Katedra informatiky
Přírodovědecká fakulta
Univerzita Palackého v Olomouci

BAKALÁŘSKÁ PRÁCE

Software pro anotaci sekvencí a GO analýzu diferenciálně exprimovaných genů

2016

Mgr. Filip Kokáš

Vedoucí práce: Mgr. Tomáš Kühr, Ph.D.

Studijní obor: Aplikovaná informatika, kombinovaná forma

Bibliografické údaje

Autor:	Mgr. Filip Kokáš
Název práce:	Software pro anotaci sekvencí a GO analýzu diferenciálně exprimovaných genů
Typ práce:	bakalářská práce
Pracoviště:	Katedra informatiky, Přírodovědecká fakulta, Univerzita Palackého v Olomouci
Rok obhajoby:	2016
Studijní obor:	Aplikovaná informatika, kombinovaná forma
Vedoucí práce:	Mgr. Tomáš Kühr, Ph.D.
Počet stran:	51
Přílohy:	1 CD/DVD
Jazyk práce:	český

Bibliograhic info

Author:	Mgr. Filip Kokáš
Title:	Software for sequence annotation and GO analysis for genes with differential expression
Thesis type:	bachelor thesis
Department:	Department of Computer Science, Faculty of Science, Palacký University Olomouc
Year of defense:	2016
Study field:	Applied Computer Science, combined form
Supervisor:	Mgr. Tomáš Kühr, Ph.D.
Page count:	51
Supplements:	1 CD/DVD
Thesis language:	Czech

Anotace

Bakalářská práce pojednává o vývoji softwaru pro anotaci nukleotidových a proteinových sekvencí s následnou analýzou zaměřenou na GO termy u diferenciálně exprimovaných genu. Program je určen pro bioinformatickou analýzu, která navazuje na celotranskriptomové sekvencování, prováděné za účelem detekce diferenciálně exprimovaných genů. Teoretický přehled je zaměřen na relevantní skutečnosti pro bioinformatickou analýzu nukleotidových a proteinových sekvencí a na sekvenační technologie. Práce rovněž obsahuje popis testování programu na reálných datech.

Synopsis

Bachelor thesis discusses the development of software for annotation of nucleotide and protein sequences followed by analysis focused on the GO terms for differentially expressed genes. The program is designed for bioinformatics analysis that follows the whole-transcriptional sequencing for detection of differentially expressed genes. Theoretical review focuses on the relevant facts for the bioinformatics analysis of nucleotide and protein sequences and technologies of sequencing. Work also contains a description of the testing program on real dataset.

Klíčová slova: bioinformatika, analýza diferenciálně exprimovaných genů, GO termy, anotace sekvencí, fasta format

Keywords: bioinformatics, analysis genes with differential expression, GO terms, sequence annotation, fasta format

Děkuji panu Mgr. Tomáší Kührovi, Ph.D. za odborné vedení a cenné rady v průběhu vypracování a sepisování bakalářské práce.

Místopřísežně prohlašuji, že jsem celou práci včetně příloh vypracoval/a samostatně a za použití pouze zdrojů citovaných v textu práce a uvedených v seznamu literatury.

datum odevzdání práce podpis autora

Obsah

Seznam obrázků

Seznam tabulek

1 Úvod a cíle práce

Na počátku tisíciletí došlo k mohutnému rozvoji sekvenačních technologii. Řada technik umožňujících efektivní sekvencování genomu a transkriptomu byla vyvinuta v prvním desetiletí 21. století. Paralelně s těmito technikami dochází rovněž k rozvoji oblasti bioinformatiky zabývající se zpracováním informací produkovaných v průběhu sekvenačního experimentu. Nezbytnou součástí bioinformatické analýzy je získaným sekvencím přiřadit jejich funkci v zájmu lepšího pochopení vztahu mezi získanými experimentálními daty a fyziologickými projevy zkoumaného organismu. Kromě řady experimentálních metod, které lze použít pro tento účel, je velmi častým přístupem prohledávání veřejných nukleotidových nebo proteinových databází, kde jsou podobné sekvence již anotovány. Hledání a vytváření nástrojů schopných efektivně prohledávat veřejné databáze, za účelem získávání těchto informací, představuje důležitý krok v bioinformatické analýze.

Práce si klade za cíl vytvoření programu, který bude schopen efektivně prohledávat veřejné nukleotidové a proteinové databáze a výsledky ukládat do vlastní vytvořené databáze, za účelem pozdější analýzy. Program rovněž bude umožňovat provádět bioinformatickou analýzu výsledků celotranskriptomového sekvenování se zaměřením na detekci diferenciálně exprimovaných genů, s využitím programem vytvořené databáze s informacemi o sekvencích. Analýza bude prováděna relativním porovnáním vlastností sekvencí mezi anotovaným souborem sekvencí a jeho podmnožinou reprezentovanou sekvencemi se změněnou mírou exprese. Výstup z analýzy bude uživateli poskytnut v tabulkovém formátu.

Bakalářská práce je rozčleněna do několika kapitol. První z nich v sobě zahrnuje úvod do relevantní problematiky, která souvisí s vývojem a používáním programu. Druhá kapitola obsahuje programátorskou příručku, ve stručnosti dokumentující strukturu vytvořeného programu. Třetí kapitolou je uživatelská příručka, která poskytuje potenciálnímu uživateli relevantní skutečnosti, jak a pro jaké účely program užívat. Poslední kapitolou je část dokumentující testování programu na reálných datech.

2 Teoretický úvod do problematiky

V této kapitole jsou uvedeny základní pojmy relevantní pro vytvářený program. Dále je zde zahrnut stručný popis fasta formátu a sekvenačních technologii. Poslední část kapitoly se zabývá anotací nukleotidových a proteinových sekvencí.

2.1 Druhy sekvencí a základní pojmy

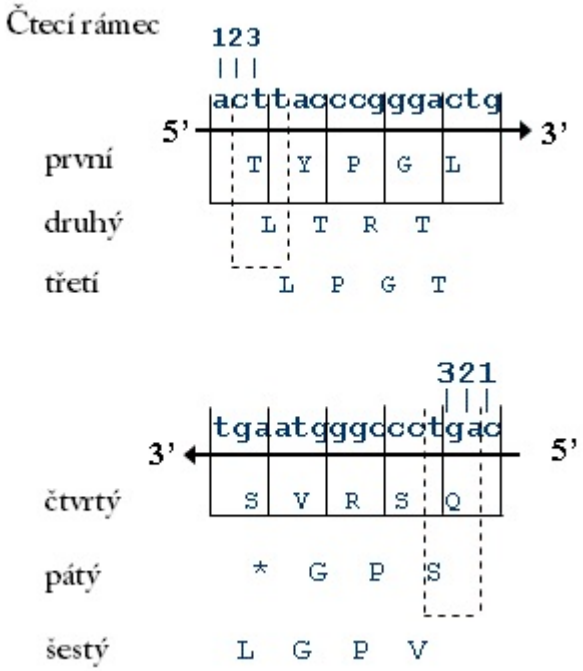
Rozlišujeme dva základní typy sekvencí, a sice nukleotidové a aminokyselinové, někdy nazývané také proteinové. V nukleotidových sekvencích se vyskytují nukleotidy reprezentované jednopísmennými zkratkami A,T,G,C nebo U reprezentující adenin, thymin, guanin, cytosin, respektive uracil.

Tabulka 1: Třípísmenné a jednopísmenné aminokyselinové zkratky s odpovídajícími triplety RNA. V DNA je uracil (U) nahrazen thyminem (T; modifikováno dle [Nirenberg, 2004]).

Třípísmenná zkratka	**Jednopísmenná zkratka**	**Nukleotidové triplety**					
Ala	A	GCA	GCC	GCG	GCU		
Arg	R	AGA	AGG	CGA	CGC	CGG	CGU
Asp	D	GAC	GAU				
Asn	N	AAC	AAU				
Cys	C	UGC	UGU				
Glu	E	GAA	GAG				
Gln	Q	CAA	CAG				
Gly	G	GGA	GGC	GGG	GGU		
His	H	CAC	CAU				
Ile	I	AUA	AUC	AUU			
Leu	L	UUA	UUG	CUA	CUC	CUG	CUU
Lys	K	AAA	AAG				
Met	M	AUG					
Phe	F	UUC	UUU				
Pro	P	CCA	CCC	CCG	CCU		
Ser	S	AGC	AGU	UCA	UCC	UCG	UCU
Thr	T	ACA	ACC	ACG	ACU		
Trp	W	UGG					
Tyr	Y	UAC	UAU				
Val	V	GUA	GUC	GUG	GUU		
Stop kodon	-	UAA	UAG	UGA			
Start kodon	-	AUG					

V DNA každého organismu, která je tvořena nukleotidovými sekvencemi, lze nalézt entity zvané geny. Jedná se o úseky řetězce DNA se specifickou funkcí, které jsou v průběhu transkripce přepisovány do RNA. Některé z nich jsou pak nadále přepisovány do aminokyselinové sekvence tvořící protein. Úsek DNA, který bude podroben transkripci je charakterizován tzv. „start“ a „stop“ kodonem, které ohraničují začátek, respektive konec transkribovaného úseku DNA. Kodon je tvořen třípísmennou sekvencí nukleotidů tzv. tripletem **Tab.1**. Triplety následně určují, které aminokyseliny budou tvořit protein, který je produkován v průběhu translačního procesu. Výsledný protein je charakterizován aminokyselinovou sekvencí, která je tvořena jednopísmennými, případně třípísmennými zkratkami aminokyselin (podrobněji viz. **[Alberts, 1998]**).

Nukleotidová sekvence podrobena transkripci je přepsána do RNA, která má několik typů. Souhrn všech molekul RNA ve zkoumané buňce tvoří transkriptom. Pro sekvencování transkriptomu je ve většině případů důležitá pouze mediátorová RNA (mRNA), kterou lze charakterizovat jako RNA, která je následně podrobena translaci, a tudíž je z ní produkován protein. V genomu organismu se nachází velké množství genů, ze kterých vznikají proteiny. Pokud je provedeno sekvencování všech molekul mRNA, které se nacházejí ve zkoumané buňce, jedná se o celotranskriptomové sekvencování.



Obrázek 1: Čtecí rámce pro příklad nukleotidové sekvence, s vyznačenými jednopísmennými zkratkami odvozených aminokyselin. *, stop kodon; **[NCBI]**.

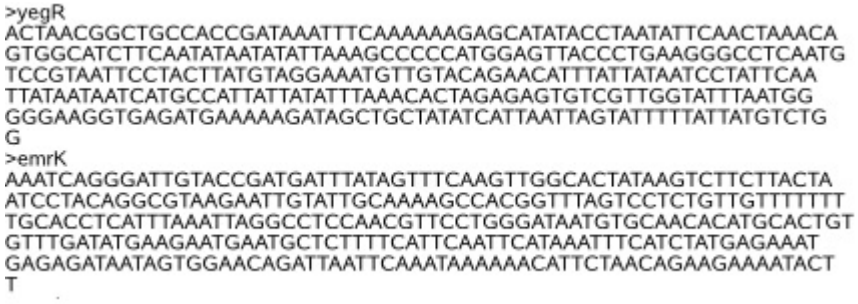
Kvantifikováním počtu kopii mRNA stejného genu je získaná hodnota vyjadřující počet kopii mRNA daného genu ve zkoumané buňce. Tuto hodnotu lze definovat jako míru exprese zkoumaného genu.

V některých případech je ovšem vyžadováno provést srovnání buněk, které jsou vystaveny různým podmínkám, nebo se nacházejí v různých tkáních, případně v různém čase. Touto problematikou se zabývá diferenciální transkriptomika. Dle očekávání, v takto zkoumaných buňkách probíhá transkripce a následná translace různých genů. Geny, u kterých je zaznamenán rozdíl v míře genové exprese nazýváme diferenciálně exprimované geny.

Nukleotidová sekvence může být podrobena translaci a výsledná proteinová sekvence může vzniknout na základě celkem šesti čtecích rámců (**Obr.1**). Čtecí rámec definuje způsob, jakým lze číst nukleotidovou sekvenci a přepisovat ji do sekvence aminokyselinové. Z celkového počtu celkem šesti možných čtecích rámců, jsou tři z nich ve směru 5´ - 3´ a další tři jsou ve směru 3´ - 5´ . Označení 3´ a 5´ definují orientaci molekuly DNA z hlediska volných konců molekuly deoxyribózy tvořící uhlíkovou kostru DNA (podrobněji viz. **[Alberts, 1998]**). Na základě použitého čtecího rámce může tedy vzniknout až šest možných proteinových sekvencí. Skutečnost, která z těchto sekvencí, bude nejpravděpodobněji přepisována do proteinu, je do jisté míry odvoditelná, na základě počtu „stop“ kodonů a přítomnosti „start“ kodonu v daném čtecím rámci (**Tab.1**).

2.2 Fasta formát

Fasta je textový formát, který je v bioinformatice určen pro reprezentaci nukleotidových nebo proteinových sekvencí. Datový soubor obsahující sekvence ve formátu fasta, jsou velmi často označovány příponou .fasta nebo .fa. Datový formát byl nejprve implementován v rámci softwaru primárně určeného pro porovnávání nukleotidových respektive proteinových sekvencí, publikovaném v roce 1985 **[Lipman & Pearson, 1985]**. Později se formát stal jedním ze standardních formátů využívaných pro práci se sekvencemi. Nukleotidy, případně aminokyseliny, jsou zde reprezentovány nejčastěji ve formě jednopísmenných zkratek (**Tab.1**).



Obrázek 2: Ukázka fasta formátu pro zapisování sekvencí.

Samotné sekvence, lze definovat jako konečné textové řetězce jednopísmenných znaků reprezentujících aminokyselinu nebo nukleotid. Kromě samotné nukleotidové nebo proteinové sekvence je v souboru rovněž umístěn pro každou sekvenci jeden řádek obsahující jméno sekvence, případně její podrobnější popis (**Obr.2**). V rámci fasta souboru obsahujícím velké množství sekvencí slouží tento řádek zároveň jako identifikátor sekvence před kterou je uveden.

2.3 Sekvenační technologie

Historie sekvencování má své počátky v roce 1977, kdy byla poprvé publikována Sangerova a Maxam-Gilbertova metoda sekvencování ([**Maxam & Gilbert, 1977**], [**Sanger *et al.*, 1977**]). Obě tyto technologie náleží mezi sekvenační metody první generace. V průběhu následujících desetiletí, prošly metody sekvencování první generace několika vylepšeními. Mezi ty nejpodstatnější lze uvést inovaci pomocí kapilární elektroforézy [**Swerdlow & Gesteland, 1990**] nebo zavedení metody fluorescenčního značení nukleotidů [**Smith *et al.*, 1986**].

Postupem času se začaly vyvíjet nové metody, umožňující cenově dostupnější sekvencování obrovského množství fragmentů DNA s větší efektivitou [**Tucker *et al.*, 2009**]. Tyto metody byly později nazvány sekvenačními metodami druhé generace. Metody v této skupině poskytují obrovské množství krátkých sekvencí nukleotidů („ready“). Počty takto získaných krátkých sekvencí se mohou pohybovat od několika stovek tisíců pro sekvencování DNA bakterii, až po jednotky miliard pro velké sekvenační projekty jako je sekvencování DNA pšenice. Popisované metody sekvencování lze využít pro celou řadu aplikací, mezi které náleží celogenomové sekvencování, sekvencování transkriptomu (RNASeq), cílené resekvencování nebo hledání bodových mutací u zkoumaného organismu ([**Schendure & Ji, 2008**];[**Metzker, 2009**]).

K sekvenačním technologiím druhé generace náleží metoda Roche/454, založena na pyrosekvencování [**Ronaghi *et al.*, 1996**], metoda Illumina využívající sekvenaci syntézou [**Turcatti *et al.*, 2008**], nebo metoda SOLID založená na sekvencování ligací [**Schendure *et al.*, 2008**].

Pro sekvencování transkriptomu je ovšem u většiny používaných technologii nutné provést nejprve přepis sekvence mRNA zpět do sekvence DNA. Tento postup je definován jako reverzní transkripce a dochází při něm ke vzniku tzv. cDNA. Takto získaná cDNA je následně s pomocí specifických enzymů fragmentována a sekvenována zvolenou sekvenační metodou. Výsledkem procesu sekvencování transkriptomu, je velmi často jeden, případně dva soubory obsahující „ready“ vznikající sekvencováním fragmentů cDNA. Délka sekvenovaných fragmentů může být různá na základě zvolené technologie a procesu fragmentace cDNA. Fragmenty cDNA mají dva konce. Na základě zvolené technologie je možné provádět sekvencování pouze z jednoho konce (sekvencování typu single-end), nebo z obou konců (sekvencování typu paired-end). Při sekvencování paired-end je získáno větší množství informací, které mohou následnou bioinformatickou analýzu uči-

nit přesnější. Výsledné soubory jsou ve formátu FASTQ, který je variací formátu fasta [Wang *et al.*, 2009].

2.4 Analýza výsledků sekvencování

Postup analýzy je velmi úzce spjat se skutečností, zda se jedná o sekvencování celogenomové nebo sekvencování transkriptomové. V případě celogenomového sekvencování je dalším postupem rekonstrukce genomu s využitím nástrojů podobných pro analýzu transkriptomu přístupem de novo. Pokud se jedná o resekvencování genomu (sekvencování genomu u kterého je již známa sekvence DNA), lze volit postup podobný jako u transkriptomu, ovšem v závislosti na tom, za jakým účelem je sekvencování prováděno.

V případě sekvencování transkriptomu jsou možné tři analytické přístupy následné bioinformatické analýzy. Prvním přístupem je *de novo*, kdy není k dispozici referenční genom. Mezi programy používanými pro tento způsob analýzy, patří zejména Trans-AbySS [Robertson *et al.*, 2010], Oases [Schultz *et al.*, 2012] nebo Trinity [Grabherr *et al.*, 2011].

V rámci druhého přístupu k analýze transkritptomu je prováděno mapování na referenční genom (přístup *ab initio*). U tohoto přístupu se využívá skutečnost, že „ready“ ze sekvenátoru jsou, co se týče sekvence, shodné s referenčním genomem. Programy tedy vyhledají shodu a spárují sekvenci „readu“ se sekvencí reference. V současné době je k dispozici více než 60 programů, které se touto problematikou zabývají ([Lang *et al.*, 2012];[Fonseca *et al.*, 2012]). Výhodou této strategie oproti přístupu *de novo*, je transformace problému sestavování transkriptomu do mnoha menších problému, které jsou z hlediska časové náročnosti snadněji řešitelné. V procesu je rovněž ve velké míře využívána paralelizace výpočtů [Martin & Wang, 2011]. Mezi programy zabývající se mapováním na referenční genom patří zejména GNUMAP [Clement *et al.*, 2010], BFAST [Homer *et al.*, 2009], TopHat [Trapnell *et al.*, 2009] nebo GSNAP [Wu & Nacu, 2010].

Třetím způsobem je kombinovaný přístup zahrnující obě možnosti [Martin & Wang, 2011]. Obecně platí zásada, že pokud má experimentátor k dispozici referenční genom, volí se přístup s pomocí mapování nebo kombinovaný přístup. U řady nemodelových organismů, kdy referenční genom ještě není k dispozici není ovšem jiná možnost než se spolehnout na přístup *de novo* [Duan *et al.*, 2012].

Pokud chceme provádět diferenciální analýzu transkriptomu, pro který není k dispozici referenční genom, je možné využít možnost přístupu *de novo*, s pomocí kterého nejprve vytvoříme referenční genom *in silico*. Při tomto postupu jsou „ready“ spojovány do sekvencí, které pravděpodobně v buňce představují DNA genů. S takto vytvořeným referenčním genomem lze již postupovat přístupem *ab initio*.

Po dokončení procesu mapování je v případě transkriptomu provedena kvantifikace „readů“, velmi často také s následnou analýzou diferenciální genové ex-

prese. Proces kvantifikace se snaží zjistit jaký počet kopii mRNA genů zkoumaného organismu, se nacházel ve zkoumané buňce. Pro tento proces je naprosto klíčové vědět, kde se v genomu zkoumaného organismu geny nacházejí, tedy jaké jsou pozice jejich nukleotidových sekvencí v rámci celé sekvence genomové DNA. Potřebné informace jsou obsaženy v anotačním souboru, který rovněž obsahuje další informace a genech zkoumaného organismu. Formát anotačního souboru je gtf, ale lze se setkat i s formáty gff2 nebo gff3. Informace obsažené v anotačním souboru se odlišují hlavně svou strukturou, v závislosti na zvoleném formátu, nicméně společné informace, které jsou v těchto souborech uloženy, zahrnují informace o začátku a konci genu v referenčním genomu. V procesu kvantifikace namapovaných „readů“, jsou velmi často využívány programy jako je HTSeq [**Anders *et al.*, 2014**], Cuffdiff [**Trapnell *et al.*, 2010**] nebo BEDTools [**Quinlan & Hall, 2010**]. Získaná data jsou následně podrobena normalizaci a s pomocí statistické analýzy je provedena identifikace diferenciálně exprimovaných genů (DEGs; [**Osklack *et al.*, 2010**]). Normalizace je prováděna pro následné správné srovnání úrovně exprese jednotlivých genů mezi dvěma vzorky.

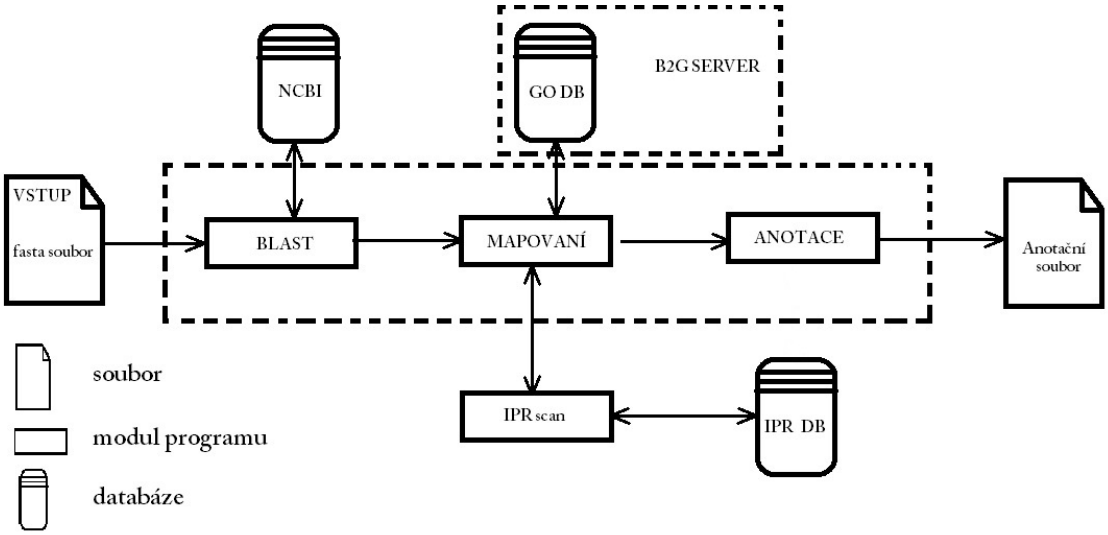
Mezi často prováděné normalizace patří normalizace dat na délku genu nebo normalizace dat na celkový počet „readů“ pro zkoumané vzorky [**Osklack *et al.*, 2010**]. Velmi používanou metodou normalizace je metoda RPKM („Reads Per Kilobase per Million mapped reads“), která provádí normalizaci z hlediska velikosti genové sekvence i z hlediska velikosti celého souboru namapovaných „readů“ [**Mortazavi *et al.*, 2008**]. Další možnost normalizace představuje metoda FPKM („Fragments Per Kilobase of exon per Million mapped reads“), která se využívá především pro sekvencování typu paired-end [**Trapnell *et al.*, 2010**].

Pro detekování diferenciálně exprimovaných genů, jsou využívány volně dostupné softwary, mezi které náleží Cufflinks [**Trapnell *et al.*, 2010**], DESeq [**Anders & Huber, 2010**] nebo edgeR [**Robinson *et al.*, 2010**]. Programy velmi často využívají různé statistické testy [**Seyednasrollah *et al.*, 2015**]. Mezi dnes nejvíce využívané programy patří DeSeq a edgeR, především z důvodu poskytování nejlepších výsledků v rámci publikovaných srovnávacích studií [**Soneson & Dolorenzi, 2013**]. Obvyklým výstupem z analýzy diferenciální exprese je tabulka obsahující seznam genů společně s číselnými údaji charakterizující míru diferenciální exprese.

2.5 Anotace sekvencí

Anotace sekvencí tvoří důležitou součást bioinformatické analýzy diferenciálně exprimovaných genů a všech dalších analýz zahrnujících pochopení vztahu mezi fenotypem organismu a přítomností zkoumaných sekvencí v genomu respektive v transkriptomu.

Počet sekvencí, které je třeba anotovat, se velmi často pro transkriptomy pohybuje v rozsahu několika tisíců i desetitisíců. Vzhledem k tomu, je v hojné míře využívána automatizace celého procesu [**Grabherr *et al.*, 2011**]. Principem anotace je identifikace vlastností sekvence na základě podobnosti s jinými sek-



Obrázek 3: Schéma programu Blast2GO s vyznačenými jednotlivými kroky a spojení s veřejnými databázemi a servery. GO DB, databáze Gene Ontology; IPR DB, databáze InterPro; IPR scan, InterPro scan (přepracováno dle [Conesa *et al.*, 2005]).

vencemi uloženými ve veřejných databázích, které již mají požadované vlastnosti definovány [Wit *et al.*, 2012].

Jeden z programů využívaný pro anotaci nukleotidových nebo proteinových sekvencí je Blast2GO. Jedná se o program nezávislý na použité počítačové platformě vyžadující pouze internetové připojení [Conesa *et al.*, 2005]. Průběh anotačního procesu lze v programu Blast2GO rozdělit do tří izolovaných kroků nacházející se v sériovém uspořádání (Obr. 3).

První krok v anotačním procesu je reprezentován procesem „blastování“ využívající program BLAST (Basic Local Alingment Search Tool). Program zajišťuje vyhledávání vstupních, neboli dotazovaných, sekvencí nukleotidů nebo aminokyselin dodávaných programu jako vstup ve formátu fasta (viz podkapitola 2.1). Podle toho, zdali se jedná o aminokyselinovou nebo nukleotidovou sekvenci, rozlišujeme několik nastavení ve kterých může program BLAST pracovat (Obr. 4).

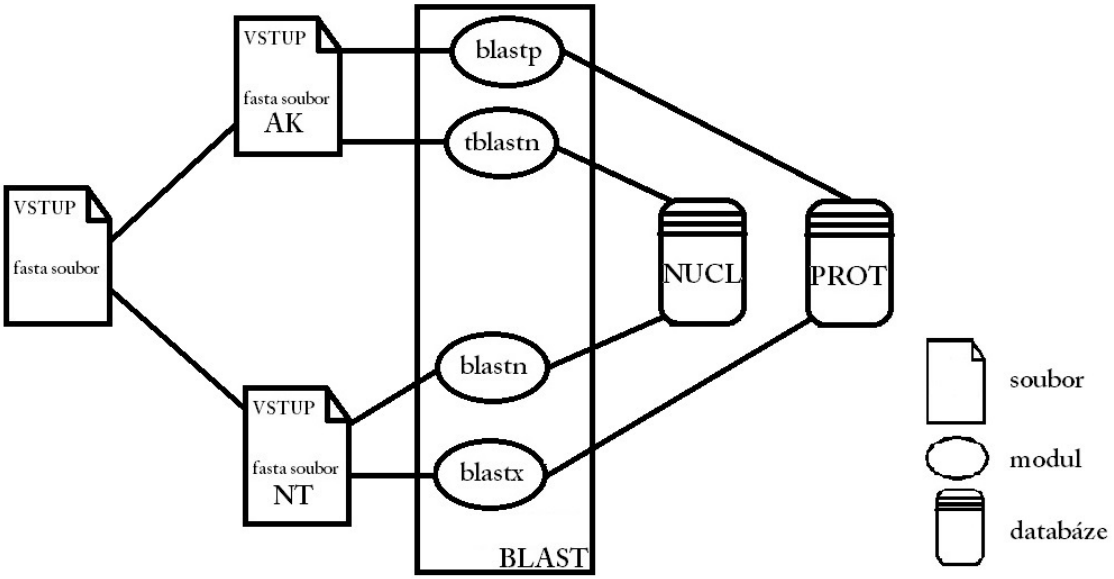
První možností je režim **blastn**, který je určen výlučně pro nukleotidové sekvence. Analogicky při prohledávání aminokyselinových sekvencí je využíváno nastavení **blastp**. Mezi další nastavení, které uživatel programu může zvolit, patří **blastx**, kdy je vstupem nukleotidová sekvence a dochází k prohledání všech šesti možných čtecích rámců prohledávané nukleotidové sekvence. Program umožňuje pracovat ještě v režimu **tblastn**, ve kterém dochází k blastování nukleotidových sekvencí, získaných zpětným převodem aminokyselinové sekvence na nukleotidovou (dle Tab.1; [Camacho *et al.*, 2009]; [Camacho *et al.*, 2013]).

Před samotným spuštěním programu lze navolit další parametry zabývající

se především kvalitou získaných výsledků a volbou databáze, kterou bude program prohledávat. Velmi důležitým parametrem je rovněž hodnota **e-value** popisující šum pozadí a tedy kvalitu získaného výsledku blastování. Čím více se hodnota **e-value** blíží nulové hodnotě, tím větší je pravděpodobnost, že podobnost zkoumané sekvence se sekvencí nalezenou je skutečná a nikoliv pouze náhodná [McGinnis & Maddden, 2004]. Nejčastější volba databáze, je „nt“ (Nucleotide Collection) pro nukleotidové sekvence, případně „nr“ (Non-redundant protein sequences) pro sekvence proteinové.

Druhým krokem procesu anotace sekvencí pomocí programu Blast2GO je tzv. mapování. Jedná se o proces získávání „Gene Ontology“ (GO) termů na základě výsledků poskytnutých procesem blastování [Conesa & Gotz, 2008].

GO termy představují vlastnosti příslušné sekvence. Každý GO term je charakterizován jedinečným alfanumerickým kódem, který slouží jako identifikátor. GO term rovněž obsahuje název, výčet případných synonym a definici, která specifikuje vlastnost, která je GO termem reprezentována. GO termy pro zkoumanou sekvenci poskytují informace o tom, kde se nachází případný translační produkt (protein), jeho molekulární funkci a biologický proces, kterého se účastní [Diehl *et al.*, 2007]. Informace o GO termech jsou uloženy v databází Gene Ontology (GO databáze). Údržbu GO databáze zajišťuje „Gene Ontology Consortium“ [Gene Ontology].



Obrázek 4: Vybrané možnosti práce programu BLAST se vstupními sekvencemi na základě uživatelem definovaných parametrů a druhu prohledávaných sekvencí. NT, nukleotidová sekvence, AK, aminokyselinová sekvence; NUCL, nukleotidová databáze; PROT, proteinová databáze (zpracováno na základě [Camacho *et al.*, 2009]; [Camacho *et al.*, 2013]).

Do procesu mapování je v poslední době zahrnován také InterPro scan (IPR scan). Jedná se o program provádějící analýzu proteinových sekvencí a poskytující výsledky, díky kterým, je proteinová sekvence klasifikována z hlediska zařazení do skupin na základě mnoha různých kritérii. Tento proces zahrnuje komunikaci s veřejnou databází IPR, která obsahuje informace o proteinových sekvencích, které již byly anotovány ([Mitchel *et al.*, 2015]; [Jones *et al.*, 2014]).

Posledním krokem anotačního procesu je samotný anotační krok, při kterém dochází k přiřazování pravděpodobných funkčních GO termů sekvencím ze souboru GO termů nashromážděných v mapovacím kroku [Conesa & Gotz, 2008].

3 Programátorská příručka

Kapitola je rozčleněna do několika částí. Nejprve jsou stručně charakterizovány použité technologie. Ve druhé podkapitole je nastíněn popis strategie použité při implementaci programu. V poslední části, je vysvětlena struktura databáze a možnosti dalšího vývoje programu.

Samotný program se zabývá analýzou diferenciálně exprimovaných genů z hlediska znalosti anotačních informací o těchto genech. Poskytuje uživateli možnost vytvoření databáze, ve které budou uloženy informace o sekvencích, které byly získány pomocí programu blast a IPR scan (viz **podkapitola 2.4**). Veškeré takto získané informace program zpracuje, na základě informací poskytnutých z analýzy diferenciální genové exprese (viz **podkapitola 2.3**). Uživateli program poskytuje výstup v podobě tabulkového formátu, který zobrazuje změněné vlastnosti reprezentované GO termy pro diferenciálně exprimované geny ve vztahu k celému souboru prohledávaných genů. V programu je možné provádět i některé další operace jako je „update“ databáze, nebo její vymazání které jsou podrobněji popsány v uživatelské příručce.

Vytvořený program ve své činnosti sleduje velmi podobný postup v průběhu anotace sekvencí jako program Blast2GO, který je popisován v teoretické části. Při anotaci sekvencí je stejně jako v případě programu Blast2GO vstupem fasta soubor obsahující prohledávané sekvence. Následně dochází k blastování sekvencí ze vstupního souboru a k vyhledávání dalších informací s pomocí IPR scanu. Tyto dva procesy je program schopen provádět paralelně ve srovnání s programem Blast2GO, kde je IPR scan prováděn až v rámci kroku „mapování“ (**Obr. 3**).

Program rovněž nekomunikuje se vzdáleným serverem B2G (viz **Obr. 3**), ale informace, které jsou relevantní pro pozdější analýzu a správnou interpretaci GO termů si stahuje ze serveru GeneOntology. Blast2GO rovněž neobsahuje možnost zpracovávat data z programu DESeq jako vstup pro následnou analýzu GO termů ve zkoumaném souboru genů (viz **podkapitola 4.2**). Tuto skutečnost se snaží program vyvíjený v rámci této bakalářské práce vyřešit a urychlit tak bioinformatickou analýzu diferenciálně exprimovaných genů.

V průběhu samotné analýzy implementovaný program na základě GO termů spočítá jejich obsah pro geny, které jsou diferenciálně exprimované a provede relativní porovnání s celým souborem zkoumaných genů.

Při implementaci programu byl rovněž kladen důraz na možnost, že může být pracováno s obrovskými objemy dat, a z toho důvodu byl zvolen databázový systém MySQL pro jejich uložení a následné operace s nimi. Při práci s velkými objemy vstupních dat tedy nebude docházek v programu k nestandardním stavům jako tomu je u programu Blast2GO, kdy tato skutečnost byla ověřena při práci s velkými objemy dat.

Vývoj programu probíhal ve spolupráci s Oddělením molekulární biologie Centra regionu Haná pro biochemický a biotechnologický výzkum a jeho funkce byly implementovány na základě konzultace s potenciálními uživateli.

3.1 Použité technologie

Program byl vytvořen s pomocí kombinace programovacího a databázového jazyka. Prvním z nich, je programovací jazyk Perl [Perl], který zajišťuje uživatelské rozhraní, komunikaci s webovými službami zajišťujícími prohledávání veřejných databází a případný import/export dat. Databázový jazyk SQL řeší správu a uložení dat. Program byl vytvořen pro databázový systém MySQL [MySQL] a je primárně cílen na linuxový operační systém distribuce Ubuntu [Ubuntu].

Vývoj probíhal na platformě s operačním systémem Linux distribuce Ubuntu verze 14.04. Testování bylo prováděno na zařízení, kde byl program vyvíjen a na serveru s operačním systémem linux distribuce Ubuntu 12.13.

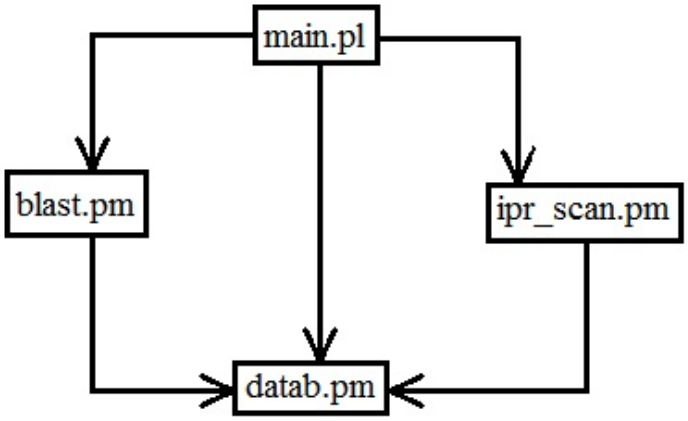
Pro svou činnost program vyžaduje nainstalovaný interpret jazyka Perl a databázový systém MySQL. Samotný vývoj aplikace probíhal na platformě obsahující interpret jazyka Perl v5.18.2 a databázovým systémem MySQL v5.5.47. Program dále pro svou činost vyžaduje následující moduly jazyku Perl, které jsou dostupné na serveru CPAN (Comprehensive Perl Archive Network; [CPAN]):

- threads
- Term::ANSIColor
- LWP
- English
- XML::Simple
- XML::XPath
- File::Basename
- Getopt::Long
- Data::Dumper
- Term::ReadKey
- MySQL::Backup (verze 0.04)

3.2 Popis implementace programu

Program je na základě volby parametru schopen provádět několik možných akci, které budou podrobněji popsány v uživatelské dokumentaci. Funkce, které program využívá, jsou rozčleněny do několika modulů. V rámci modulu jsou funkce rovněž členěny do skupin funkcí z důvodu snadnější orientace. Všechny funkce jsou ve zdrojovém kódu opatřeny svým krátkým popisem pomocí komentářů vložených do zdrojového kódu, a rovněž pomocí identifikátoru, který je složen ze dvou částí, a sice z několika písmenné zkratky vyjadřující skupinu do které

je funkce zařazena a číselný identifikátor, který identifikuje funkci v rámci skupiny. Následující text popisuje jednotlivé vytvořené moduly a skupiny funkcí v nich obsažené. Struktura vytvořeného programu z hlediska modulů je zobrazena v diagramu (**Obr. 5**).



Obrázek 5: Diagram zobrazující moduly programu a vzájemné reference ve zdrojovém kódu.

3.2.1 Soubor main.pl

Soubor main.pl obsahuje uživatelské rozhraní programu. Uživatel jeho prostřednictvím spoští program. Soubor obsahuje zejména programem definované nastavení některých parametrů, které uživatel má ovšem možnost změnit s pomocí údajů zadaných do terminálu při spuštění (viz **uživatelská dokumentace**). Kód zde obsažený rovněž provádí na základě zadaných parametrů spouštění požadovaných aktivit, které program vykonává, obsluhu běhu programu v průběhu jeho činnosti a následně korektní ukončení programu. Jsou zde obsaženy reference na vytvořené moduly, které jsou klíčové pro správné fungování programu.

Kromě výše uvedeného je v tomto souboru zahrnuta jedna skupina funkcí, která byla pojmenována MAIN a sdružuje v sobě funkce zajišťující zobrazení nápovědy a obsluhu vláken určených pro BLAST a IPR scan.

3.2.2 Modul blast.pm

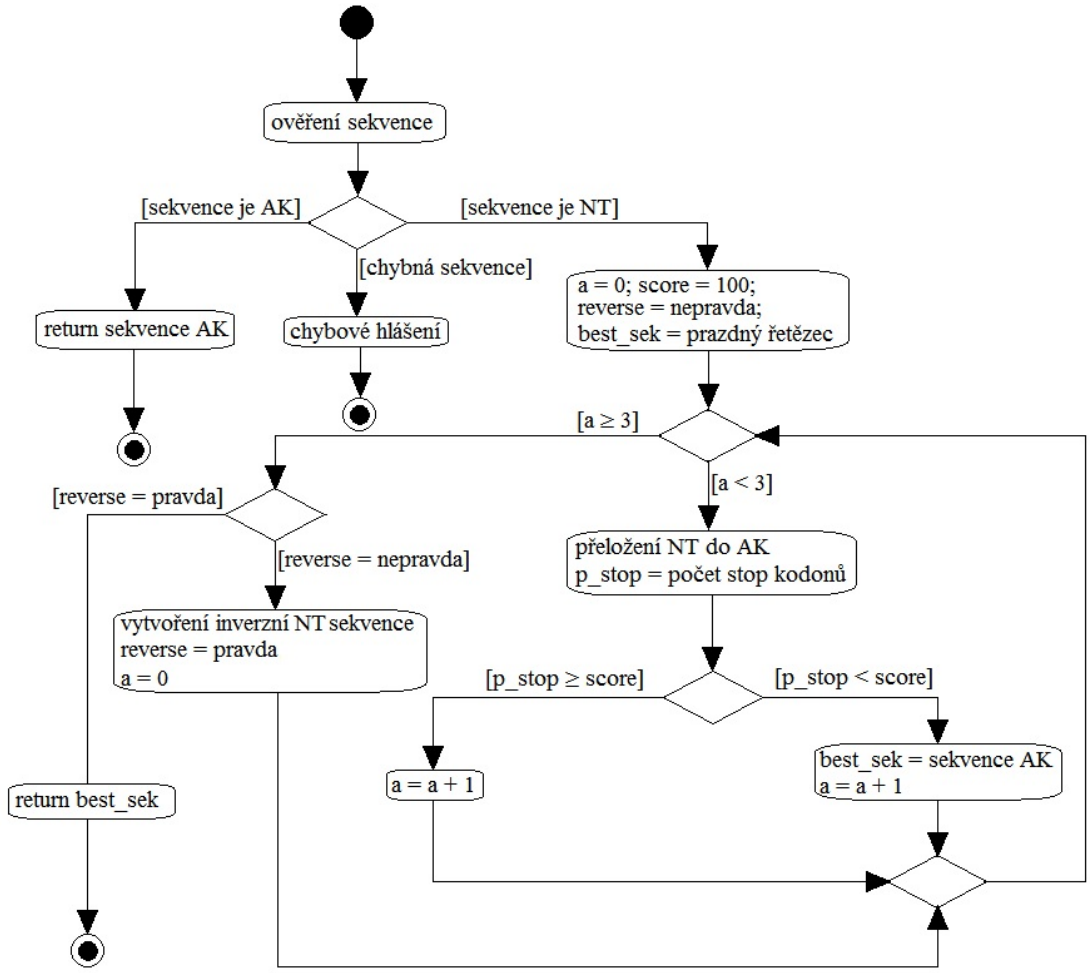
První modul Perlu obsahuje jedinou skupinu funkcí s názvem BLAST. Dle názvu se tato skupina funkcí zabývá procesem odeslaní blastované sekvence na vzdálený server NCBI společně s parametry nutnými pro správné provedení procesu (viz **uživatelská dokumentace** a **kapitola o blastovani** v teoretické části). V modulu jsou rovněž vyžadovány některé funkce z modulu datab.pm, pro následné uložení získaných výsledků do programem vytvořené databáze MySQL.

3.2.3 Modul ipr_scan.pm

Druhý z vytvořených modulů zajišťuje úkoly obdobné pro modul blast.pm, ovšem s rozdílem, že pro prohledávání sekvence v rámci InterPro scanu je nutné, aby prohledávaná sekvence byla aminokyselinová. Modul je rozčleněn do tří skupin funkcí, zajišťujících specifické úkoly.

První skupina je tvořena funkcemi CONTROL. Jedná se o skupinu zajišťující kontrolu, zda sekvence, kterou uživatel vkládá do programu splňuje vlastnosti aminokyselinové, případně nukleotidové sekvence (viz **podkapitola o sekvencích** v teoretické části).

Druhá skupina funkcí s názvem CONV v tomto modulu, zajišťuje konverzi nukleotidové sekvence do sekvence aminokyselinové. Tento úkol zahrnuje určení nejlepšího čtecího rámce u nukleotidové sekvence a samotnou konverzi na sekvenci aminokyselinovou (**Obr. 6**).



Obrázek 6: Diagram zobrazující algoritmus vyhledání nejlepšího čtecího rámce. AK, aminokyselinová; NT, nukleotidová

Algoritmus určení nejlepšího čtecího rámce je zahájen ověřením sekvence. V průbehu tohoto kroku je na základě aminokyselinových/nukleotidových zkratek (viz **Tab. 1**) určeno, zdali se jedná o sekvenci aminokyselinovou nebo nukleotidovou. Pokud v sobě vstupní řetězec obsahuje neznámé znaky, je vypsáno chybové hlášení. V případě, kdy výsledkem ověření sekvence je skutečnost, že se jedná o aminokyselinovou sekvenci je tato sekvence vrácena jako výsledek algoritmu a použita pro prohledávání pomocí IPR scanu (viz **podkapitola 4.2**).

Pokud se jedná o sekvenci aminokyselinovou je nastavena hodnota „score“ reprezentující vhodnost sekvence pro prohledávání IPR scanem. Čím menší je hodnota „score“ tím je sekvence vhodnější. Další hodnotou která je zadefinována je proměná „a“, která reprezentuje čtecí rámec, který je posuzován algoritmem (**Obr. 6**). Vzhledem k tomu, že čtecích rámců je celkem 6 (3 v přímém a 3 v opačném směru; viz **podkapitola 2.1**) hodnota proměnné „a“ nabývá hodnot 0,1 a 2 podle skutečnosti, který čtecí rámec je prohledáván. Při prohledávání čtecího rámce v přímem směru je parametr „reverse“ nastaven na hodnotu nepravda a při prohledávání čtecích rámců ve zpětném směru je nastaven na hodnotu pravda (**Obr. 6**).

Vhodnost čtecího rámce je posuzována na základě počtu „stop“ kodonů ve zkoumané sekvenci a jeho srovnáním s hodnotou proměnné „score“. Jako výsledek algoritmu jehož vstupem je nukleotidová sekvence je vrácena aminokyselinová sekvence s nejmenším počtem „stop“ kodonů. „Start“ kodony v tomto algoritmu hodnoceny nejsou, vzhledem ke skutečnosti, že „start“ kodony velmi často nejsou vůbec v sekvenci uváděny a ve srovnání se „stop“ kodony, jsou pro hodnocení vhodného čtecího rámce téměř nepodstatné.

Poslední a nejrozsáhlejší skupinou, jsou funkce IPR. Zde obsažené funkce mají podobný význam jako v modulu blast.pm, ovšem pro IPR scan. Rovněž tato skupina funkcí využívá některé funkce modulu datab.p, pro zdárné ukladání výsledků prohledávání do databáze (**Obr. 5**).

3.2.4 Modul datab.pm

Jedná se o nejrozsáhlejší modul tohoto programu. Modul v sobě sdružuje funkce zajišťující ukládání dat do databáze, případně jejich export, za účelem získání textového výstupu v podobě „tabulky“ v txt/csv souboru. Databáze je vytvářena v databázovém systému MySQL **[MySQL]** a obsahuje informace podstatné pro činnost programu (podrobněji **podkapitola 3.3**). Veškeré funkce jsou v rámci modulu datab.pm rozděleny do celkem pěti skupin.

První skupina s názvem BASIC zahrnuje základní funkce, klíčové pro správu databáze. Patří sem zejména řešení přístupu k databázi, vytvoření databáze, její struktury a některé další základní příkazy, které je možné provést. Mezi ně náleží, zejména zobrazení existujících databází, vymazání zvolené databáze nebo vymazání dat ze zvolené tabulky.

Druhá skupina funkcí s názvem INSDB obsahuje funkce zajišťující import dat do databáze, především pro BLAST a IPR scan. Rovněž jsou v této skupině

obsaženy funkce pro importování uživatelem vkládaných údajů o Gene Ontology termech nebo o diferenciální expresi zkoumaných genů.

Třetí skupinu tvoří analytické funkce (skupina ANF) pro analýzu diferenciálně exprimovaných genů z hlediska obsahu Gene Ontology termů. Funkce z této skupiny jsou využívány především, u nastavení analysis (viz **uživatelská dokumentace**).

Čtvrtá skupina funkcí pro tento modul (skupina s názvem EXP) zajišťuje export databázových tabulek do výstupního txt formátu, na základě požadavku uživatele, v rámci nastavení export (viz **uživatelská dokumentace**).

Poslední skupinu tvoří funkce, které zajišťují výpis varování, upozornění nebo informací poskytovaných programem uživateli v průběhu výpočetního procesu do výstupního souboru s názvem log.txt.

3.3 Struktura databáze

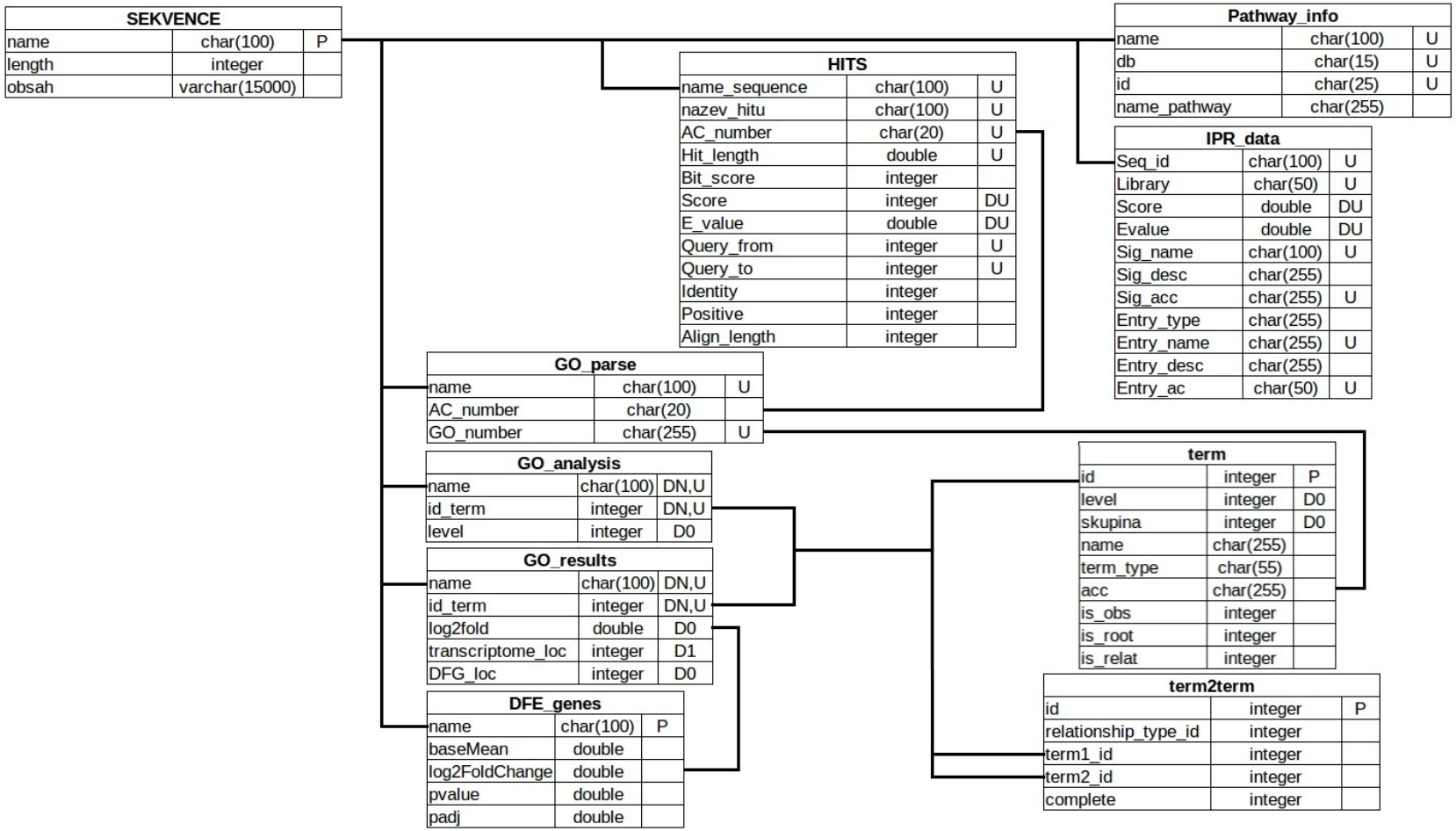
Program v rámci tvorby databáze vytváří 10 databázových tabulek (**Obr. 7**). Tabulky s názvy term a term2term obsahují informace o významu (tabulka term) a vzájemných vazbách (tabulka term2term) mezi jednotlivými GO termy (podrobněji viz **příloha B**).

Informace jsou do těchto tabulek vkládány z databáze MySQL, kterou spravuje Gene Ontology Consorcium. Databáze je denně aktualizována. Z toho důvodu bylo zvoleno, aby program při každé tvorbě další databáze, tyto informace stahoval znovu, aby byla zajištěna aktualita dat, se kterými program pracuje. Pokud je prováděna analýza s využitím již vytvořené databáze, tak informace o GO termech aktualizovány nejsou a tabulky term a term2term jsou považovány za dostatečně aktuální. V tabulce term jsou přítomny dva sloupce, které nejsou součástí původní tabulky. Jedná se o sloupce s názvem level a skupina, které mají svůj význam, především v nastavení analysis při analýze diferenciálně exprimovaných genů.

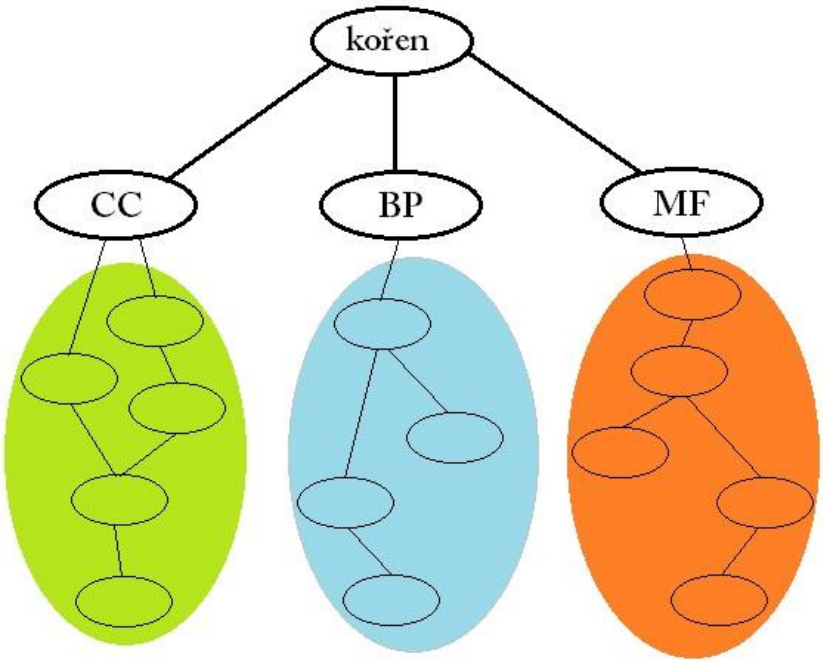
Tabulka s názvem SEKVENCE, uchovává základní informace o prohledávaných sekvencích, které uživatel vkládá do databáze prostřednictvím fasta souboru. Pro práci programu jsou uvedeny informace, jako je jméno, délka sekvence a samotná vkládaná nukleotidová nebo proteinová sekvence.

Výsledky procesu blastování jsou ukládány do tabulky HITS a stejně tak výsledky z IPR scanu jsou ukládány do tabulek IPR_data a Pathway_info. Podrobnější informace o tabulkách a vysvětlení významu jednotlivých sloupců je zobrazena v **příloze B**. Informace o diferenciálně exprimovaných genech, které uživatel vkládá ve formě csv souboru, jsou ukládány do tabulky DFE_genes.

Jak blastovaní, tak i IPR scan produkují záznamy o GO termech. Záznamy z obou procesů jsou ukládány do tabulky GO_parse. GO termy se vyznačují definovanou strukturou vzájemných vztahů, kterou lze reprezentovat v podobě grafu (**Obr. 8**). Graf GO termů má jediný hlavní vrchol a tři hlavní podgrafy mezi kterými neexistují, žádné spojující hrany. Jednotlivé podgrafy jsou nazývány, na základě GO termů tvořících jejich nejvýše položený vrchol, který je nejblíže



Obrázek 7: Schema databáze MySQL vytvořené programem. DU – default null; P – primary key; N – not null; D0 – default 0; D1 – default 1.

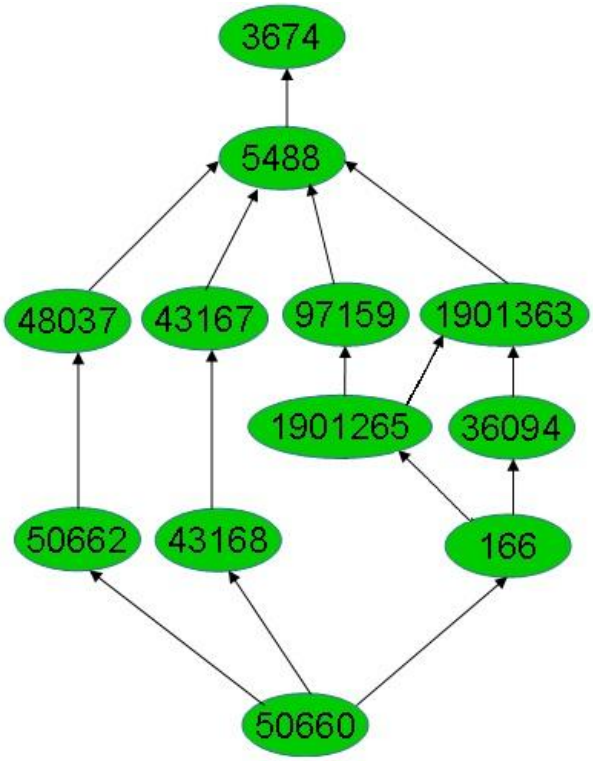


Obrázek 8: Struktura GO termů reprezentovaná ve formě grafu, s vyznačením podgrafů. CC, Cellular Component; BP, Biological Process; MF, Molecular Function

hlavnímu vrcholu. Jedná se o podgrafy „Biolological Process“ (BP), „Molecular Function“ (MF) a Cellular Component (CC). Vrcholy podgrafů BP, CC a MF již mohou mít více hran vedoucích k níže položeným vrcholům.

Mezi těmito vrcholy, již mohou existovat vzájemné hrany (**Obr. 9**). V průběhu provádění analýzy diferenciálně exprimovaných genů, jsou GO termy použité pro analýzu rozděleny do tří kategorii na základě jejich příslušnosti k danému podgrafu. V databázi je tato skutečnost pro účely analýzy uložena ve sloupci „skupina“ v tabulce „term“. Nejvýše položený vrchol celého grafu kategorii nemá a z toho důvodu je jeho kategorie označována jako „Unknown“ (UN).

U jednotlivých GO termů je rovněž pro účely pozdější analýzy počítána tzv. úroveň („level“). Jedná se o hodnotu zobrazující, kolik vrcholů leží v nejkratší cestě mezi zkoumaným GO termem a nejvyšším vrcholem grafu, včetně nejvyššího vrcholu a vrcholu reprezentujícího příslušný GO term (**Obr. 9**). Informace je pro účely analýzy uložena ve sloupci „level“ v tabulce „term“. V tabulce s názvem „GO_parse“ jsou ovšem obsaženy pouze GO termy pro příslušné geny z nejnižších vrstev podgrafu, pro které platí, že vlastnost charakterizovaná GO termem nejvíce odpovídá skutečné vlastnosti genu. Pro následnou analýzu je ovšem zapotřebí znát veškeré GO termy z vyšších vrstev, které příslušejí danému genu. Pokud platí, že gen mající vlastnost, která je reprezentovaná GO termem „A“ a tomuto termu je ve struktuře grafu nadřazen GO term „B“, pak také platí, že gen má vlastnost „B“. Za účelem získání těchto vlastností pro ná-



Obrázek 9: Příklad části orientovaného grafu zobrazující GO termy, s vyznačením identifikátorů GO termů a vzájemných vazeb mezi GO termy. GO termy, jsou reprezentovány alfanumerickými identifikátory, používanými pro GO termy (přepracováno podle databáze Gene Ontology Consortium; **Gene Ontology**). 50660, Vazba flavin adenin dinukleotidu; 50662, Vazba koenzymu; 43168, Vazba aniontu; 166, Vazba nukleotidu; 1901265, Vazba nukleosid fosfátu; 36094, Vazba malých molekul; 48037, Vazba kofaktoru; 43167, Vazba iontu; 97159, Vazba cyklické organické sloučeniny; 1901363, Vazba heterocyklické organické sloučeniny; 5488, Vazba sloučeniny; 3674, Molekulární funkce (MF).

slednou analýzu je využita tabulka „GO_analysis“, do které jsou uloženy informace o všech GO termech příslušejících zkoumaným genům, určené na základě struktury grafu. Takto získané informace jsou následně použity pro analýzu prováděnou programem.

Některé z databázových tabulek nejsou využity v následné analýze, ani nenacházejí využití, při dalších činnostech programu. V průběhu vývoje, bylo rozhodnuto o uložení těchto dat do databáze, navzdory skutečnosti, že nejsou klíčové pro analýzu prováděnou programem. Tento postup byl zvolen, zejména pro možnost, pozdějšího rozšiřování programu a následnou implementaci nových přístupů pro analýzu získaných dat.

4 Uživatelská příručka

Kapitola se zabývá popisem programu z pohledu potenciálního uživatele a dokumentuje operace, které může uživatel s programem provádět. Kapitola je rozčleněna do několika částí. První část představuje parametry, které jsou programem akceptovány, a jejich přípustné, případně programem definované, hodnoty. Druhá část představuje jednotlivé nastavení programu, jeho očekávané vstupy a výstupy. Poslední část krátce představuje možnosti monitorování činnosti programu.

4.1 Parametry programu

Program se ovládá prostřednictvím terminálu, zadáváním kombinace uživatelem volitelných a programem vyžadovaných parametrů. Příkaz pro spouštění programu v terminálu operačního systému Linux vypadá následovně:

main *–**db_user** uživatel –**mode** text* [*parametry*]

Parametry lze rozdělit do dvou skupin. První skupinu parametrů, jsou ty, které jsou vyžadovány při každém spouštění programu. Mezi parametry tohoto druhu náleží „db_user“, „db_host“ a „mode“ (**Tab. 2**). První dva z těchto parametrů jsou klíčové pro spojení mezi programem a databázovým serverem MySQL. V programu je hodnota parametru „db_host“ přednastavena na hodnotu „localhost“. Pokud tedy přednastavená hodnota souhlasí s tou, kterou má v úmyslu zadat uživatel, není nutné tuto hodnotu uvádět. Třetím parametrem je parametr „mode“. S pomocí něj, uživatel specifikuje, jakou činnost bude program vykonávat. Na základě hodnoty tohoto parametru jsou určovány i další parametry, které program vyžaduje pro svou činnost v daném nastavení (**Tab. 2**; viz **podkapitola 4.2**). Posledním z parametrů, které je nutné vždy zadat je parametr „db_heslo“ (**Tab. 2**). Tento parametr se ovšem nevpisuje přímo do terminálu při spouštění programu, ale je po uživateli vyžádán bezprostředně po jeho spuštění. Při zadávání hesla nejsou z důvodu bezpečnosti vidět vkládané znaky.

Druhá skupina parametrů je tvořena takovými parametry, které jsou vyžadovány pouze při určité činnosti programu, na základě nastavení parametru „mode“ (**Tab. 2**).

Speciální postavení v této skupině má parametr „help“ (**Tab. 2**). Při jeho zadání, program vypíše stručnou nápovědu pro své použití a svou činnost ukončí. Stejného efektu je rovněž docíleno při zadání nesprávné kombinace parametrů.

Pro některá nastavení je nutné definovat některé další parametry. Zejména v případě analýzy se mohou stát velmi užitečnými parametry „pAdj_cut“ a „log_cut“. Parametr „pAdj_cut“ představuje hodnotu s jakou pravděpodobností je gen diferenciálně exprimován. Pokud je hodnota nižší než 0.05, gen s touto hodnotou „pAdj_cut“ je diferenciálně exprimován z pravděpedobností větší než 95%.

V případě parametru „log_cut“ je situace složitější. U vkládaných záznamů, se předpokládá, že obsahují hodnotu pro „log2FoldChange“, která udává míru

Tabulka 2: Parametry vyžadované programem, jejich přednastavené a možné hodnoty a jejich nutnost výskytu, v závislosti na nastavení parametru „mode“. A – parametr je při daném nastavení vyžadován; M – uživatel má možnost tento parametr zadat což může částečně změnit činnost programu.

Název	**Význam**	**Defaultní hodnota**	**Možné hodnoty**	**Nastavení proměnné mode**									
				create_only	create	update	analysis	insert	delete	show	migrate	export	check
db_user	Uživatel databázového systému MySQL		text/číslo	A	A	A	A	A	A	A	A	A	A
db_host	Host server	localhost	text/číslo	A	A	A	A	A	A	A	A	A	A
db_heslo	Heslo do databázového systému MySQL		text/číslo	A	A	A	A	A	A	A	A	A	A
db_name	Jméno databáze		text/číslo	A	A	A	A	A	A		A	A	A
mode	Nastavení programu	analyze	hodnoty proměnné mode	A	A	A	A	A	A	A	A	A	A
blast_mode	Nastavení vyhledávání BLAST	blastn	blastn/blastp/ blastx/tblastn		A	A							
db_blast	Název databáze pro BLAST	nt	nt/nr/pdb		A	A							
email	E-mai		e-mail		A	A							
help	Parametr pro zobrazení nápovědy												
input	Vstupní soubor		cesta k souboru		A	A	M	A	A		M	M	
out_format	Výstupní formát souboru txt		txt/csv/-				A					A	
out_pref	Prefix pro výstupní soubory	out	text				A					A	
e_value	Hodnota e-value pro filtrování výsledků BLAST prohledávání	0.001	číslo		A	A							
max_count_hit	Maximální počet výsledků pro BLAST	20	číslo		A	A							
pAdj_cut	Minimální hodnota pAdj pro diferenciálně exprimované geny	0.01	číslo				A						
log2f_cut	Minimální/maximální hodnota pro up/-down regulované geny	0.5	číslo				A						

diferenciální exprese (viz **podkapitola 2.1**). Na základě toho, jestli je hodnota kladná nebo záporná je zkoumaný gen up-regulován nebo down-regulován. Pokud je tedy parametr „log_cut“ nastaven na hodnotu 2.0, pak jsou všechny geny, které mají hodnotu „log2FoldChange“ uvedenou v intervalu -2.0 až 2.0 z následné analýzy vyřazeny.

Parametry „pAdj_cut“ a „log_cut“ tedy určují podmínky analýzy. Pokud je například hodnota parametru „pAdj_cut“ nastavena na 0.05, pak všechny záznamy o genech, které mají hodnotu pAdj vyšší než 0.05 nebudou brány v průběhu analýzy do úvahy.

Pokud program v rámci své činnosti provádí prohledávání sekvencí s pomocí programů BLAST a IPR scan, pak parametry nutné pro tuto činnost jsou zejména „email“, „db_blast“ a „blast_mode“. První parametr je vyžadován programem IPR scan, ostatní dva parametry jsou vyžadovány programem BLAST (viz **podkapitola 2.5**). Dalšími důležitými parametry v procesu prohledávání sekvencí, jsou parametry „max_count_hit“ a „e_value“. První parametr určuje, jaký maximální počet výsledků získaných blastováním bude uloženo do databáze. Druhý parametr, představuje filtr kvality výsledků a tedy blastovací výsledky s hodnotou „e_value“ vetší než stanovená hranice nebudou do databáze ukládány, ani vyžívány pro následné vyhledávání GO termů. Oba tyto parametry jsou přednastaveny programem (**Tab. 3**).

4.2 Činnost programu

Na základě parametru „mode“ má uživatel možnost řídit činnost programu. Sledování činnosti programu je kromě příkazové řádky zapisováno do souboru log.txt (viz **podkapitola 4.3**). Uváděný parametr může nabývat následujících hodnot.

a) create_only

V tomto nastavení dojde k vytvoření databáze společně se všemi tabulkami a je provedeno naplnění databázových tabulek informacemi obsahující údaje o GO termech. Konkrétně se jedná o tabulky s názvy „term“ a „term2term“, jejichž obsah je z velké části získáván ze serveru Gene Ontology (viz **podkapitola 3.3**). Samozřejmostí je uvedení parametru „db_name“ pro následnou identifikaci databáze v rámci databázového systému.

Při tomto nastavení nedochází k prohledávání databází a proto není třeba dávat programu vstupní soubor se sekvencemi ve formátu FASTA. U tohoto nastavení ovšem dochází k dopočítání vrstev v rámci stromové struktury GO termů (viz **podkapitola 3.3**). Rovněž dochází k rozdělení GO termů na základě jejich výskytu v podstromech MF, BP nebo CC (viz **podkapitola 3.3**).

b) create

Tato možnost zahrnuje všechny náležitosti „create_only“, ovšem s následným zpracováním vstupních sekvencí programy BLAST a IPR scan. Proces

blastování probíhá na základě uživatelského nastavení příslušných parametrů (Tab. 2). Následně je proveden výpočet rodičovských GO termů pro zkoumané sekvence, na základě struktury GO termu ve formě grafu (viz podkapitola 3.3).

c) update

Uživatel požaduje v některých případech provést aktualizaci již vytvořené databáze, nebo provést přidání několika sekvencí do již existující databáze. Pro tyto případy je vytvořeno nastavení „update“. Program v tomto případě projde všechny záznamy o sekvencích uložených v databázi v tabulce „SEKVENCE“ a u těch, které v databázi neobsahují informace o GO termech, provede opětovné vyhledávání s pomocí programu blast a IPR scan.

Pokud chce uživatel do databáze zahrnout nové sekvence, s pomocí parametru „input“ lze do databáze přidat sekvence, které budou následně prohledávány standardním způsobem. Pokud jsou vkládané sekvence již v databázi obsaženy, není provedeno jejich nové prohledávání. Pro sekvence, které dosud nejsou v databázi obsaženy je provedeno jejich přidání do databáze. Veškeré nalezené duplicity nejsou, vzhledem k vytvořené databázové struktuře, do databáze ukládány.

d) insert

Pokud uživatel požaduje přímé vkládání informací o GO termech získaných jiným způsobem než tímto programem, může k tomu využít tuto možnost. S pomocí parametru „input“ lze provést vložení informací o genech v podobě tabulky v textovém souboru (Tab. 3).

Jako oddělovač sloupců je předpokládán tabulátor. U souboru je rovněž předpoklad, že prvním sloupcem je identifikátor genu. Druhým sloupcem, může být buď identifikátor GO termu, nebo „AC_genu“ (viz příloha B, popis tabulky „GO_parse“), pokud se jedná o data získaná na základě výsledků blastování (příloha B). Identifikátor GO termu je třeba uvádět ve formátu GO:xxxxxxx, kde x představují čísla. „AC_genu“ představuje identifikátor záznamu získaného díky blastování. Pokud je přítomna hodnota pro „AC_genu“, pak je identifikátor GO termu uveden ve třetím sloupci. Rovněž u tohoto nastavení je prováděn výpočet rodičovských GO termů na základě jejich stromové struktury uložené v tabulkách „term“ a „term2term“.

e) analysis

Při tomto nastavení, jsou předpokládaným vstupem údaje o diferenciálně exprimovaných genech splňujících podmínky pro import do databáze (Tab. 4). Pro analýzu musí být v souboru přítomen sloupec se jmény genů jako první sloupec v souboru csv. V souboru musí rovněž být sloupce s názvem „baseMean“, „log2FoldChange“, „pvalue“ a „padj“ a jejich číselné hodnoty

Tabulka 3: Ukázka textového souboru pro vkládání informací o GO termech. První sloupec představuje název genu, druhý je identifikátor GO termu a třetí sloupec zobrazuje slovní popis názvu sekvence.

MLOC_10001	GO:0004518	hordeum vulgare vulgare mrna for complete clone: niashv1106b06
MLOC_10013	GO:0003674	
MLOC_10024	GO:0003674	triticum aestivum chromosome genomic cultivar chinese spring
MLOC_10026	GO:0003674	oryza sativa genomic chromosome bac clone: complete sequence
MLOC_1003	GO:0005975	
MLOC_10031	GO:0005975	setaria italica sucrose synthase 7-like transcript variant mrna
MLOC_10031	GO:0016757	
MLOC_10031	GO:0009058	
MLOC_10039	GO:0005737	
MLOC_10039	GO:0016798	

pro každý zkoumaný gen. Všechny tyto sloupce jsou obsaženy ve výstupu z programu DESeq2 [**Love *et al.*, 2014**].

Po importování dat ze vstupního souboru do databáze, je provedena analýza obsahu GO termů pro diferenciálně exprimované geny ve srovnání se všemi sekvencemi obsaženými v databázi. Na analýzu mají rovněž vliv parametry s názvem „pAdj_cut“ a „log2f_cut“ popsané v **podkapitole 4.1**. S pomocí parametrů „out_format“ a „out_pref“ je možné zvolit formát výstupního souboru (txt nebo csv, viz **Tab. 2**) a prefix názvu výstupních souborů.

Tabulka 4: Ukázka výstupu z programu DESeq2 ve formátu csv exportovaném do excelu [**Love *et al.*,2014**].

	baseMean	log2FoldChange	pvalue	padj
MLOC_22315	861.827	4.799	4.107e-82	7.753e-78
MLOC_18361	6399.569	-3.988	5.238e-78	4.944e-74
MLOC_36351	890.556	-3.691	3.139e-58	1.975e-54
MLOC_65804	1416.351	3.959	2.340e-57	1.104e-53
MLOC_65908	1020.166	5.270	1.657e-52	6.256e-49
MLOC_43545	1027.706	3.876	2.473e-52	7.783e-49
AT2G51510	396.7691	7.509	1.839e-51	4.961e-48
MLOC_59323	4001.651	4.912	3.053e-50	7.208e-47
MLOC_19721	919.701	5.101	8.494e-50	1.781e-46
MLOC_55855	2265.744	-2.863	1.693e-46	3.196e-43

Výstupem z analýzy jsou dva soubory, zobrazující výsledky pro up-regulované, respektive down-regulované geny (**Tab. 5**).

Název výstupního souboru je ve formátu „prefix_down_gene.format“ pro down-regulované geny a „prefix_up_gene.format“ pro up-regulované geny. Pokud uživatel zvolí pro parametr „out_pref“ hodnotu „-“ dojde k vypsání obsahu těchto souborů na standardní výstup.

Tabulka 5: Ukázka výstupu pro down-regulované geny.

Level	Group	GO_ID	Description GO term	WholeTranscriptome	DFGenes	Percentage	MeanLog2Fold
1	UN	all	all	476	109	0.2290	-2.371
2	MF	GO:0003674	molecular_function	372	89	0.2392	-2.367
2	BP	GO:0008150	biological_process	252	62	0.2460	-2.337
2	CC	GO:0005575	cellular_component	193	49	0.2539	-2.497
3	BP	GO:0051704	multi-organism process	5	2	0.4000	-3.124
3	CC	GO:0016020	membrane	63	13	0.2063	-2.334
3	BP	GO:0023052	signaling	7	3	0.4286	-1.948
3	CC	GO:0005623	cell	146	39	0.2671	-2.545
3	CC	GO:0043226	organelle	135	36	0.2667	-2.566
3	BP	GO:0000003	reproduction	3	1	0.3333	-4.413
4	CC	GO:0019867	outer membrane	1	1	1.0000	-3.988
4	BP	GO:0006950	response to stress	13	3	0.2308	-2.232
4	BP	GO:0042221	response to chemical	4	3	0.7500	-2.214
4	MF	GO:0001871	pattern binding	2	1	0.5000	-1.753

Výstupní soubory podléhají přesně definované struktuře. Jako oddělovač je zvolen tabulátor. Pokud uživatel pomocí parametru „out_format“ zvolí csv, je soubor exportován ve formě csv, kde jako oddělovač sloupců je zvolena čárka a jako oddělovače řádků je zvolen konec řádku. První sloupec zobrazuje hloubku popisovaného GO termu v rámci grafu GO termů. Druhý sloupec zobrazuje zařazení GO termu do podgrafu CC, MF nebo BP. Hodnota „UN“ značí, že daný GO term není zařazen do žádného ze tří předchozích podgrafu a tedy jeho poloha v rámci grafu není definována. Tato situace může vzniknout následkem skutečnosti, že GO term představuje nejvyšší vrchol grafu a tedy je nadřazen podgrafům CC, BP a MF. Druhou možností výskytu této skutečnosti je výskyt chyby ve výpočetním procesu,

vlivem které, není možné určit podgraf GO termu.

Sloupce 3 až 6 ve výstupním souboru zobrazují identifikátor GO termu, jeho popis, kolikrát se GO term vyskytuje u všech sekvencí v databázi, a kolikrát se vyskytuje ve zkoumané skupině diferenciálně exprimovaných genů. Předposlední sloupec zahrnuje relativní podíl GO termů mezi sloupci 5 a 6. Údaje v tomto sloupci jsou v intervalu 0 až 1. Poslední sloupec zobrazuje průměrnou hodnotu „log2FoldChange“ pro diferenciálně exprimované geny, které lze popsat vlastností charakterizovanou GO termem.

f) delete

Pokud chce uživatel některou z vytvořených databází odstranit, může k tomu využít tuto možnost. Rovněž je možnost odstranit databázi přímo v databázovém systému MySQL pomocí uživatelského rozhraní databázového systému. Uživatel je v průběhu procesu vyzván, aby potvrdil vymazání databáze. Potvrzení je možné provést pomocí zadání „y“, „Y“, „yes“ nebo „YES“. Zadání kterékoliv jiné kombinace znaků, má za následek přerušení procesu vymazání databáze.

g) show

V některých případech dojde k okolnostem, kdy má uživatel vytvořeno několik databází, ovšem ne všechny musí být vytvořeny tímto programem a mít požadovanou strukturu. Pro zobrazení databází s vyhovující strukturou dat slouží volba „show“. Výsledkem je seznam vytvořených databází splňujících požadavky programu pro práci s nimi.

h) migrate

Někdy nastává situace, kdy je třeba databázi přesunout z jednoho databázového systému do jiného. Pokud v tomto ohledu nelze využít jiného řešení například, pokud jeden z databázových systému nemá internetové připojení, je možné využít následující možnost. Pokud není v příkazu uveden parametr „input“, pak je proveden export databázových tabulek do jednoho souboru ve formě SQL příkazů.

Pokud je tento soubor následně uveden do proměnné „input“, je provedeno spuštění jednotlivých příkazu SQL a tedy vkládaní dat do databáze vytvořené programem. Databáze má název stejný jako uživatelem uvedená hodnota parametru „db_nazev“.

i) export

Pokud uživatel potřebuje s databázovými tabulkami pracovat v textovém souboru, může za účelem získání dat uložených v databázi využít toto nastavení. Data jsou exportována ve formě tabulky, kdy jako oddělovač sloupců je použit tabulátor. Tabulky, které jsou exportovány, se nazývají „HITS“ (obsahující informace o prohledávání s pomocí BLAST), „IPR_data“ (zahrnující informace získané z IPR scanu), „GO_parse“ (s GO

termy na nejnižší úrovni pro prohledávané sekvence) a tabulka „GO_analysis“ (obsahuje informace o veškerých vyhledaných GO termech vázajících se k prohledávaným sekvencím).

j) check

Jedná se o nastavení, které je velmi užitečné v případě, pokud si chceme zkontrolovat stav vytvořené databáze. Kontrolu lze provádět ovšem i v průběhu běžícího procesu prohledávání databáze, a to způsobem, kdy si uživatel otevře druhý terminál, kde spustí program s nastavením parametru mode na „check“ a s ostatními vyžadovanými parametry. Program jako výsledek vypíše do terminálu následující údaje:

In database name Test_Bc is now:

173 records in table SEKVENCE

3273 records in table HITS

33040 records in table term

33043 records in table term2term

912 records in table IPR_data

229 records in table GO_parse

0 records in table GO_analysis

0 records in table Pathway_info

In database is 40 sequences with GO records.

Výstup zobrazuje počet záznamů ve vybraných databazových tabulkách. V případě tabulky s názvem „SEKVENCE“ získáme počet sekvencí, které byly prohledány. Poslední řádek zobrazuje, pro kolik sekvencí jsou v databázi k dispozici GO termy. Počet takto uvedených sekvencí, se může značně lišit na základě okolností, jak detailně jsou již podobné sekvence anotovány.

4.3 Monitorování činnosti programu

Průběh činnosti programu je monitorován následovně. Veškeré zprávy o činnosti programu jsou vypisovány do příkazové řádky programu. Uživatel tímto způsobem může v reálném čase monitorovat činnost programu.

Kromě výpisu všech zpráv a upozornění programu do terminálu jsou zprávy o chybách, a klíčové body procesu běhu zapisovány rovněž do výstupu souboru s názvem log.txt. Zprávy, které jsou zapisovány do souboru, i zprávy vypisované v terminálu mají následující strukturu:

typ_zprávy: text_zprávy

Část „typ_zprávy“, může nabývat následujících hodnot, které určují povahu následujícího textu zprávy:

a) **ERROR** – v naprosté většině případu výpis této zprávy, znamená nestandardní ukončení programu, které může nastat zadáním chybného uživatelského vstupu nebo vlivem problému s databázovým systémem. Speciální skupinou chyb jsou ty, vyvolané problémy ve spojení se službami webu, které znemožňují získání výsledků vyhledávání v databázích. Chyby tohoto druhu nezpůsobí okamžité ukončení činnosti programu, ale chybové hlášení uživatele upozorní na problém, a že záznamy o prohledávané sekvenci nebudou uloženy do databáze. Pokud nastane chyba tohoto typu, lze pro získaní informací o neprohledaných sekvencích provést „update“ vytvořené databáze (viz. mode „update“ **podkapitola 4.2**).

b) **INFO** - zprávy tohoto typu poskytují uživateli informace o zahájení rozsáhlejšího, ve většině případů časově náročného, procesu. Mezi hlášení tohoto typu náleží zejména zahájení prohledávání vložených sekvencí nebo začátek stahovaní informací o GO termech z databáze Gene Ontology Consorcium.

c) **MESSAGE** – jedná se o zprávy tvořící pár ve vztahu ke zprávám typu INFO. Informují uživatele zejména o ukončení procesu nebo o dospění procesu do některého z klíčových bodů.

d) **WARRING** – zprávy informující uživatele, že podmínky za kterých je program spuštěn, se stávají nestandardními, ovšem nikoliv do té míry, aby vyústily v ukončení činnosti programu. Jedná se hlavně o případy, kdy je možné, že výsledkem programu bude nestandardní výstup, který může být ovšem v některých případech uživatelem odůvodnitelný.

   Řada dalších zpráv, zejména o stavu prohledávání jednotlivých sekvencí, jsou vypisovány pouze terminálem, a nejsou zapisovány do souboru log.txt, kde je prováděn zápis pouze kritických informací klíčových pro běh programu. Zprávy tohoto typu vypisují pouze text zprávy bez označení zprávy. Jedná se tedy o zprávy nízké důležitosti.

   Do souboru log.txt jsou rovněž zapisovány informace o parametrech nastavených v okamžiku spouštění programu. Pokud nastane situace, že v místě, kde je program spouštěn, již soubor log.txt existuje, jsou zprávy ze spuštěného programu připojeny za již existující obsah souboru. Uživatel tak může kontrolovat historii příkazů s pomocí kterých s programem v minulosti pracoval.

5 Testování programu na reálných datech

V této kapitole jsou popsány výsledky testování programu na reálných datech. Kapitola je rozdělena na dvě části. První část stručně popisuje vstupní data a druhá se zabývá popisem procesu a výstupů poskytnutých programem.

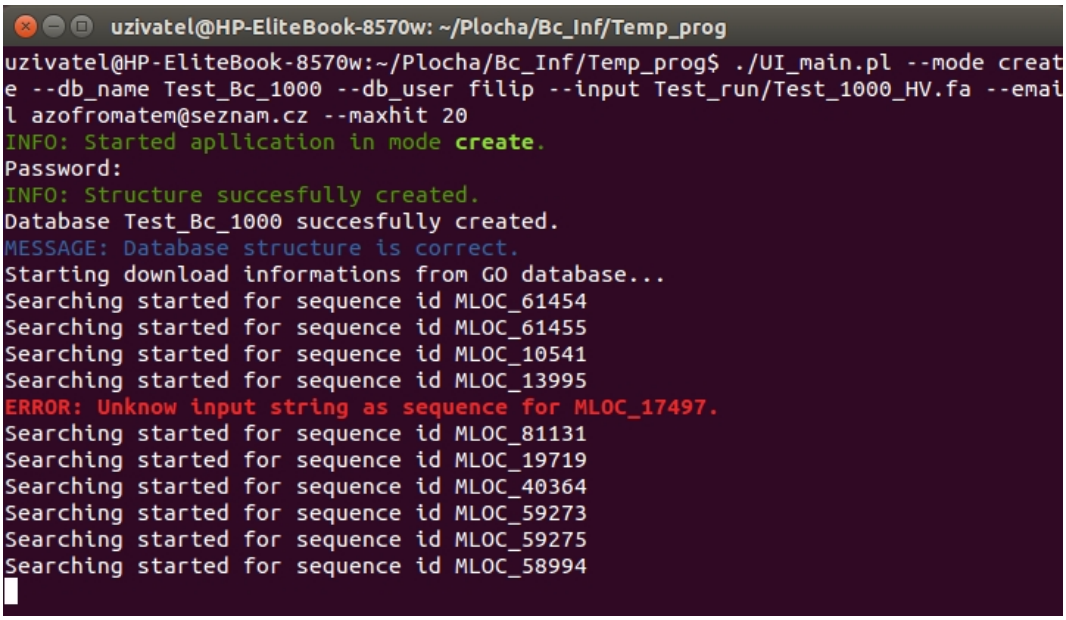
5.1 Popis vstupních dat

Datový set obsahoval 1000 sekvencí ve fasta formátu. Názvy sekvencí byly uvedeny ve formátu MLOC_xxxxx, kde x představují čísla tvořící numerický identifikátor sekvence. V datech byly zaneseny nestandardní znaky, pro sekvenci s názvem MLOC_17497. Pro účely analýzy byl sestaven soubor csv, jako výstupní soubor programu DESeq2 [Love *et al.*, 2014]. Soubor obsahoval simulované výsledky, RNAseq experimentu s následnou analýzou diferenciálně exprimovaných genů (viz **podkapitola 2.1**). Oba vstupní soubory, jsou uloženy na přiloženém CD s názvy „Test_1000_HV.fa“ a „DFG_test.csv.“

5.2 Dokumentace činnosti programu

Program byl nejprve spuštěn s pomocí příkazu:

main *–**mode** create –**input** Test_run/Test_1000.fa –**db_name** Test_Bc_1000 –**db_user** filip –**email** azofromatem@seznam.cz –**max_count_hit** 20*



Obrázek 10: Zobrazeni činnosti programu při nastavení „create“, v terminálu.

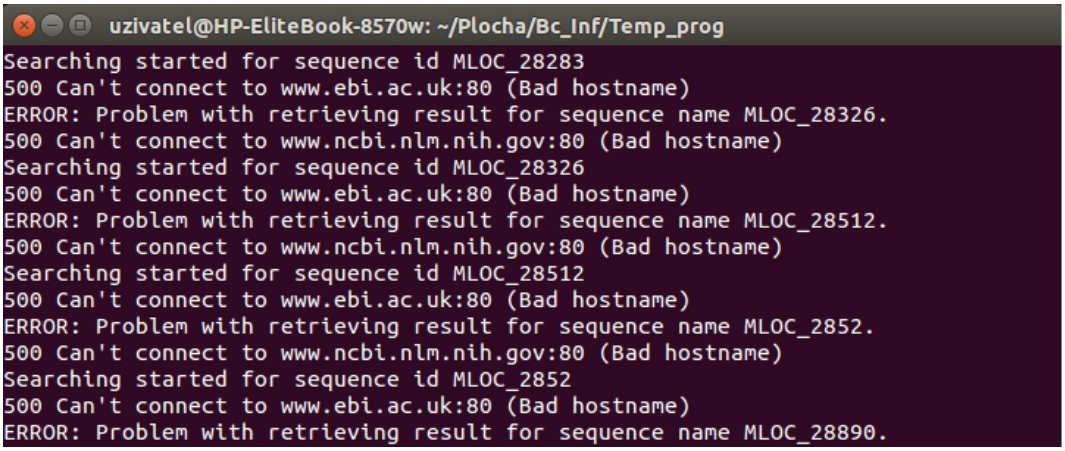
Nejprve byla v rámci činnosti programu vytvořena databáze s požadovanou databázovou strukturou (**Obr. 7**). Po vytvoření databáze následovala kontrola vytvořené databáze, zda její struktura odpovídá požadavkům programu. Uživatel byl o výsledku informován vypsáním zprávy v terminálu (**Obr. 10**).

Po kontrole databázové struktury, následovalo zahájení stahování informací z databáze GO a současně také prohledávání sekvencí obsažených ve vstupním souboru „Test_1000_HV.fa“. Program průběžně informoval o zahájení nebo ukončení prohledávání sekvence výpisem zprávy do terminálu (**Obr. 10**).

U sekvence s názvem „MLOC_17497“ byla nahlášena chyba, kdy program vyhodnotil, že vkládaná sekvence obsahuje neznámý vstupní řetězec (**Obr. 10**).

Vzhledem ke skutečnosti, že do vstupních dat byla tato chyba zanesena úmyslně, jedná se tedy o správnou reakci programu na nesprávný uživatelský vstup. Nedochází ovšem k ukončení činnosti, ale pouze k upozornění, že tato sekvence má nestandardní formu a nebude tedy nadále zpracována.

V průběhu běhu programu byl rovněž proveden experiment s odpojením internetového připojení. Výsledkem bylo, že program na základě absence internetového spojení, vypsal řadu varovných hlášení, že nelze dokončit prováděné procesy, a tedy informace, které měly být s pomocí těchto procesů získány, nelze vložit do databáze (**Obr. 11**). Program opět neukončil svou činnost nestandardním způsobem, pouze vypsal uživateli upozornění vzniklých chyb a pokračoval, ve své činnosti.



Obrázek 11: Zobrazení činnosti programu v nastavení „create“ v terminálu, při vyvolání nestandardních podmínek odpojením PC od internetového připojení.

Počítač byl po krátké době opět připojen k internetovému připojení, a činnost programu ve smyslu úspěšného prohledávání databází a následného ukládání výsledků do databáze byla úspěšně obnovena.
Zbylá činnost proběhla již bez dalších problému a program byl následně po prohledání všech sekvencí ukončen standardním způsobem. Pro kontrolu dat ulože-

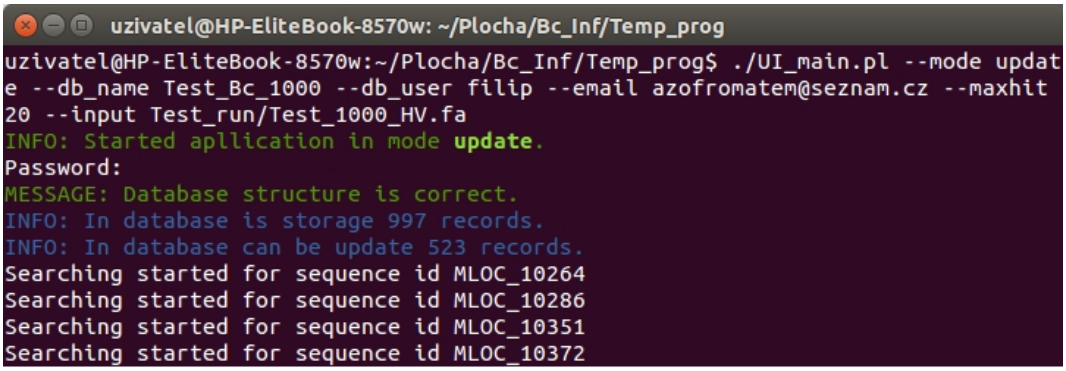
ných ve vytvořené databázi, byl program následně spuštěn v nastavení „check“. Výstup programu měl následující tvar:

In database name Test_Bc is now:
967 records in table SEKVENCE
13273 records in table HITS
44850 records in table term
91746 records in table term2term
4862 records in table IPR_data
1143 records in table GO_parse
9897 records in table GO_analysis
0 records in table Pathway_info
In database is 465 sequences with GO records.

V rámci snahy o doplnění informací o sekvencích, které nemohly být do databáze uloženy, byl program opětovně spuštěn, v nastavení „update“. Vadná sekvence ve vstupním souboru „Test_1000_HV.fa“ byla manuálně opravena. Program byl spuštěn následujícím příkazem:

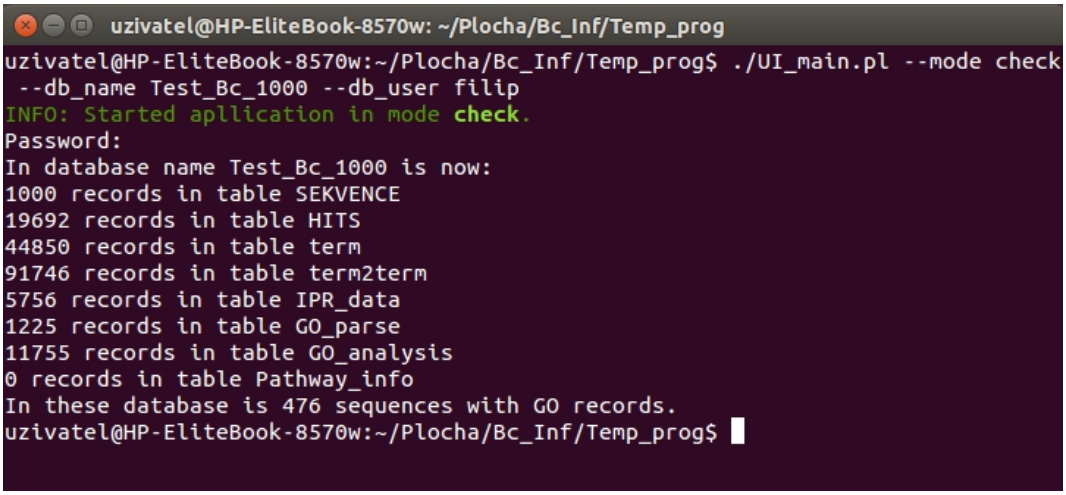
main *–**mode** update –**input** Test_run/Test_1000.fa –**db_name** Test_Bc_1000 –**db_user** filip –**email** azofromatem@seznam.cz –**max_count_hit** 20*

Program nejprve provedl kontrolu dat uložených v databázi. Na základě skutečnosti, že v databázi bylo uloženo 997 záznamů o sekvencích a o 523 sekvencí neexistují záznamy v tabulce „GO_parse“, bylo vyhodnoceno, že bude provedeno opětovné vyhledávání pro 523 sekvencí (**Obr. 12**). Samotný proces byl proveden bez hlášení problémů.



Obrázek 12: Zobrazení části činnosti programu v nastavení „update“ v terminálu.

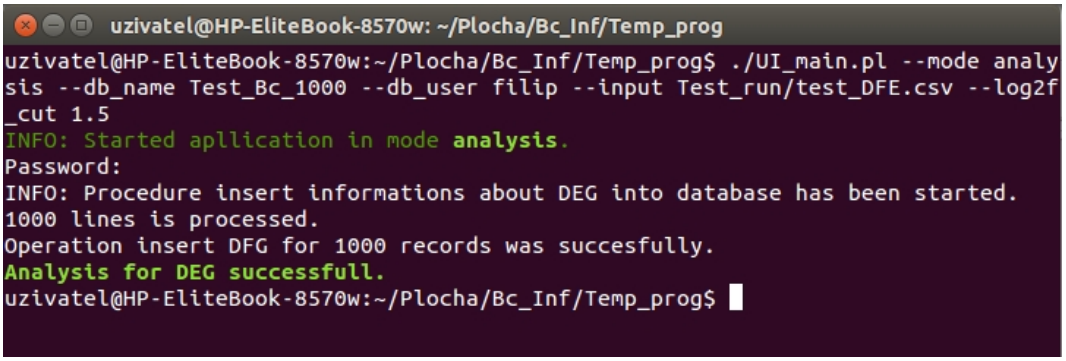
Dle předpokladu došlo k anotaci sekvencí, u který nebylo provedeno vyhledávání v databázích z výše uvedených důvodů. Dle očekávání, ovšem nedošlo k výraznému ovlivnění dat v databázi, což lze přisoudit skutečnosti, že mezi vytvořením databáze a provedením její aktualizace, uběhl poměrně malý časový úsek. Skutečnost, že došlo k anotování některých sekvencí, byla ověřena s pomocí spuštění programu s nastavením „check“ (**Obr. 13**).



Obrázek 13: Zobrazení činnosti programu v terminálu při nastavení „check“.

U vytvořené databáze obsahující údaje a sekvencích poskytnutých ve vstupním FASTA souboru, byla následně provedena analýza diferenciálně exprimovaných genů s pomocí vstupního souboru ve formátu csv, který obsahoval informace relevantní pro analytický proces. Rovněž bylo provedeno nastavení volitelného parametru „log2f_cut“ na hodnotu 1.5.

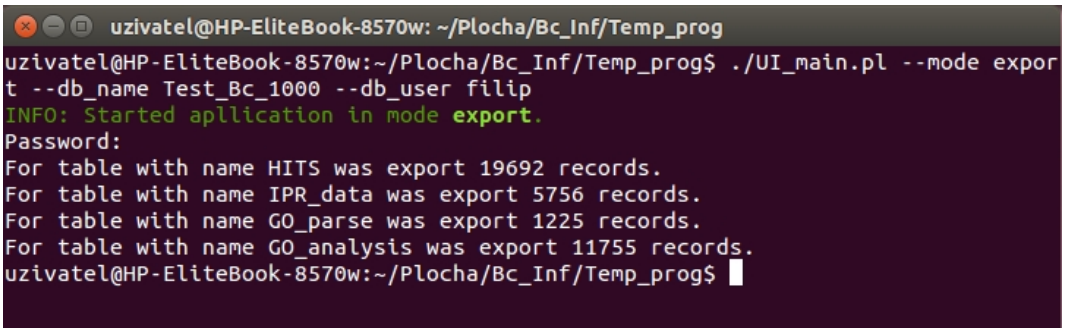
V průběhu procesu analýzy realných dat nebyly zaznamenány žádné chyby a výsledky pro up-regulované a down-regulované geny byly zaznamenány do výstupních souborů „out_down_genes.txt“ a „out_up_genes.txt“, které jsou k nahlédnutí uloženy na CD. Úspěšnost procesu, byla uživateli oznámena prostřednictvím textové zprávy v terminálu (**Obr. 14**).



Obrázek 14: Zobrazení činnosti programu v terminálu, při nastavení „analysis“.

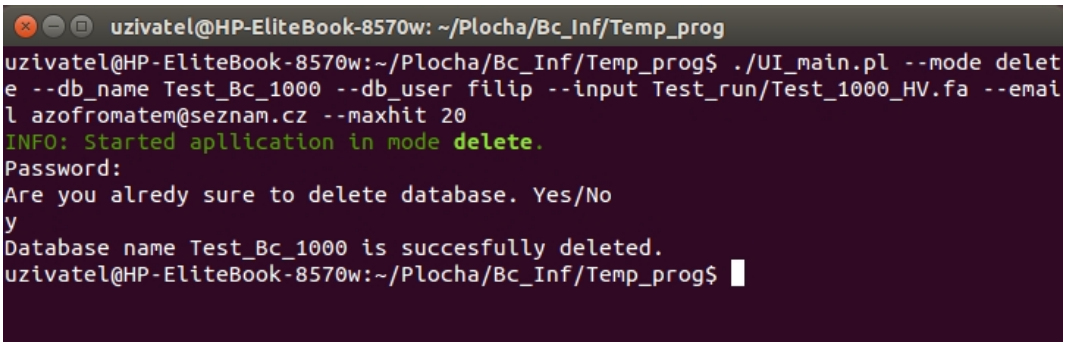
Pro případnou práci s databází ve formě textových souborů, byl proveden zkušební export dat, jehož následkem bylo vytvoření několika souborů s názvy „HITS_export.txt“ s 19 692 záznamy, „IPR_data_export.txt“ s 5 756 záznamy, „GO_parse.txt“ obsahující 1 225 záznamů a tabulka „GO_analysis.txt“ obsa-

hující v sobě 11 755 záznamů. Zároveň s vytvořením uvedených souborů, byly do terminálu vypsány informace o počtu exportovaných záznamů u příslušných tabulek (Obr. 15).



Obrázek 15: Zobrazení činnosti programu v terminálu při nastavení „export“.

Jako poslední úkon bylo v rámci testování programu provrdeno vymazání vytvořené databáze. V průběhu procesu bylo vyžadováno potvrzení procesu vymazání. Tato pojistka byla do programu zařazena především z důvodu, že vymazání databáze je nevratný krok, při kterém jsou všechna data získaná programem ztracena (Obr. 16). Databáze byla úspěšně vymazána a proces testování programu na reálných datech byl tímto ukončen.



Obrázek 16: Zobrazení činnosti programu při nastavení „delete“.

Závěr

Výsledkem této práce je program určený pro analýzu diferenciálně exprimovaných genů pomocí GO analýzy zahrnující relativní porovnání obsahu GO termů u diferenciálně exprimovaných genů a celého zkoumaného souboru genů. Program jako součást své činnosti vytváří databází MySQL a je primárně určen pro osobní počítače a servery s operačním systémem Linux distribuce Ubuntu.

Primárním výstupem programu je tabulka poskytující uživateli přehled vlastností a procesů, které byly u zkoumaného organismu ovlivněny prostřednictvím diferenciálně exprimovaných genů na základě informací uložených v programem vytvořené databázi. Program rovněž poskytuje možnost aktualizovat vytvořenou databázi nebo ji vymazat. Na základě systému zpráv poskytovaných ve formě výpisu do terminálu a v souboru log.txt rovněž umožňuje uživateli monitorování své činnosti v reálném čase.

Při testování bylo zjištěno, že je program schopen efektivně reagovat na nesprávné uživatelské vstupy a poskytuje uživateli několik možností, jak pracovat s programem vytvořenou databází. Vytvořený program může najít uplatnění především v rozsáhlých bioinformatických analýzách zaměřených na diferenciální analýzu transkriptomu, při kterých se pracuje s velkými soubory dat z hlediska počtu pozorovaných genů.

Conclusions

The result of this paper is a program designed for the analysis GO terms of differentially expressed genes. This analysis provides a comparing the relative content of GO terms for differentially expressed genes and studied the entire set of genes. Program as part of their activity creates the MySQL database and is primarily designed for personal computers and servers with the operating system Linux distribution Ubuntu.

The primary outcome of the program is a table providing overview of properties and process of investigating organism which were affected through the differentially expressed gene based on the information stored in the database. The program also provides the ability to update the database creation or delete it. On the basis of reports provided in the form of a statement to the terminal and in the log.txt file and allows the user to monitor their activity in real time.

In testing it was discovered that the program is able to respond effectively to incorrect user input and provides the user with several options of how to work with database creation program. The created program may find application in large-scale bioinformatics analyzes focused on the differential analysis of transcriptome, at which works with large set of number of observed gene.

A Obsah přiloženého CD/DVD

bin/

Adresář obsahující zdrojové kódy programu a jeho modulů, jmenovitě v souborech „main.pl“, „datab.pm“, „ipr_scan.pm“ a „blast.pm“

doc/

Text bakalářské práce ve formátu PDF, vytvořený s použitím závazného stylu KI PřF UP v Olomouci pro závěrečné práce, včetně všech příloh.

data/

Adresář obsahující testovací set dat využitý pro testování programu vytvořeného v rámci bakalářské práce. Jedná se o soubory „Test_1000.fa“ obsahující vstupní sekvence genů ve formátu fasta a soubor „test_DFE.csv“ obsahující údaje o diferenciálni expresi genů získané s pomocí programu DESeq2. V adresáři jsou rovněž obsaženy soubory vytvořené v průběhu testování programu na realných datech. Jsou zde soubory obsahující výstup z programu pro up-regulované (soubor „out_up_genes.txt“) a down-regulované (soubor „out_down_genes.txt“) geny.
Dále jsou zde obsaženy soubory vzniklé exportováním databázových tabulek. Jedná se o soubory s názvy „HITS_export.txt“, „GO_parse_export.txt“, „GO_analysis_export.txt“ a „IPR_data_export.txt“.

readme.txt

Popis instrukcí pro instalaci programu na počítač s linuxovým operačním systémem a programem vyžadovaných modulů a aplikací.

B Detailní popis databázových tabulek

Příloha obsahuje detailní popis databázových tabulek vytvářených a naplňovaných daty v rámci činnosti programu. Struktura databáze vytvářené programem je přehledně zobrazena v **obrázku 6**.

Tabulka 6: Detailní popis databázové tabulky HITS.

Název sloupce	**Formát hodnoty**	**Význam sloupce**
name_sequence	char(100)	jméno sekvence
nazev_hitu	char(100)	název hitu nalezeného blastováním
AC_number	char(20)	identifikátor hitu nalezeného blastovaním v prohledávané databázi
Hit_length	double	délka nalezeného hitu
Bit_score	integer	hodnota Bit score z výsledků prohledávaní sekvence pomocí blastování
Score	integer	hodnota score z výsledků prohledávaní sekvence pomocí blastování
E_value	double	hodnota e-value z výsledků prohledávaní sekvence pomocí blastování
Query_from	integer	srovnávaná oblast prohledávaná sekvence - počáteční nukleotid/aminokyselina
Query_to	integer	srovnávaná oblast prohledávaná sekvence - poslední nukleotid/aminokyselina
Identity	integer	identita prohledávané sekvence a hitu
Positive	integer	počet aminokyselin/nukleotidů hodnocených jako srovnatelné mezi hitem a prohledávanou sekvencí
Align_length	integer	délka srovnávaného úseku prohledávané sekvence reprezentovaná počtem aminokyselin/nukleotidů

Tabulka 7: Detailní popis databázové tabulky SEKVENCE.

Název sloupce	**Formát hodnoty**	**Význam sloupce**
name	char(100)	jméno prohledávané sekvence
length	integer	délka prohledávané sekvence (počet nukleotidů/počet aminokyselin)
obsah	varchar(15000)	prohledávaná sekvence

Tabulka 8: Detailní popis databázové tabulky GO_analysis.

Název sloupce	**Formát hodnoty**	**Význam sloupce**
name	char(100)	jméno sekvence
id_term	integer	identifikátor GO termu (tabulka term sloupec id)
level	integer	úroveň GO termu ve struktuře grafu GO termů

Tabulka 9: Detailní popis databázové tabulky IPR_data.

Název sloupce	**Formát hodnoty**	**Význam sloupce**
Seq_id	char(100)	jméno sekvence
Library	char(50)	jméno subdatabáze z databáze IPR
Score	double	hodnota score z výsledků prohledávaní sekvence pomocí ipr scanu
Evalue	double	hodnota e-value z výsledků prohledávaní sekvence pomocí ipr scanu
Sig_name	char(100)	jméno popisující výsledek získaný programem IPR scan
Sig_desc	char(255)	slovní popis záznamu získaného programem IPR scan
Sig_acc	char(255)	jednoznační identifikátor záznamu v rámci subdatabáze IPR
Entry_type	char(255)	typ záznamu spojeného s výsledkem z programu IPR scan (více specifický identifikátor než u záznamu popisovaného tabulkami Sig_name, Sig_desc, Sig_acc)
Entry_name	char(255)	jméno záznamu spojeného s výsledkem z programu IPR scan (více specifický identifikátor než u záznamu popisovaného tabulkami Sig_name, Sig_desc, Sig_acc)
Entry_desc	char(255)	slovní popis záznamu spojeného s výsledkem z programu IPR scan (více specifický identifikátor než u záznamu popisovaného tabulkami Sig_name, Sig_desc, Sig_acc)
Entry_ac	char(50)	jednoznačný identifikátor záznamu spojeného s výsledkem z programu IPR scan (více specifický identifikátor než u záznamu popisovaného tabulkami Sig_name, Sig_desc, Sig_acc)

Tabulka 10: Detailní popis databázové tabulky GO_parse.

Název sloupce	**Formát hodnoty**	**Význam sloupce**
name	char(100)	jméno sekvence
AC_number	char(20)	identifikátor hitu získaného blastováním
GO_number	char(255)	jméno GO termu (tabulka term sloupec name)

Tabulka 11: Detailní popis databázové tabulky GO_results.

Název sloupce	**Formát hodnoty**	**Význam sloupce**
name	char(100)	jméno sekvence
id_term	integer	identifikátor GO termu (tabulka term sloupec id)
log2fold	double	číselný údaj vyjadřující diferenciální expresi genu (kladná hodnota pro up-regulované geny, záporná hodnota pro down-regulované geny)
transcriptome_loc	integer	hodnota jedna pokud je záznam využit v průběhu analýzy pro celý soubor dat
DFG_loc	integer	hodnota jedna pokud je záznam využit v průběhu analýzy pro diferenciálně exprimované geny

Tabulka 12: Detailní popis databázové tabulky DFE_genes.

Název sloupce	**Formát hodnoty**	**Význam sloupce**
name	char(100)	jméno sekvence
baseMean	double	hodnota baseMean pro sekvenci z programu DESeq2
log2FoldChange	double	hodnota log2FoldChange pro sekvenci z programu DESeq2
pvalue	double	hodnota pvalue pro sekvenci z programu DESeq2
padj	double	hodnota padj pro sekvenci z programu DESeq2

Tabulka 13: Detailní popis databázové tabulky Pathway_info.

Název sloupce	**Formát hodnoty**	**Význam sloupce**
name	char(100)	jméno sekvence
db	char(15)	jméno subdatabáze z databáze IPR
id	char(25)	identifikátor metabolické dráhy
name_pathway	char(255)	jméno metabolické dráhy

Tabulka 14: Detailní popis databázové tabulky term.

Název sloupce	**Formát hodnoty**	**Význam sloupce**
id	integer	identifikátor GO termu (využito v tabulce term2term ve sloupcích term1_id a term2_id)
level	integer	úroveň GO termu v rámci struktury grafu
skupina	char(3)	identifikátor podgrafu, do kterého GO term přísluší (0 je ekvivalentní „Unknow“; 1 je „Cellular Component“; 2 je „Molecular Function“; 3 je „Biological Process“)
name	char(255)	slovní popis GO termu
term_type	char(55)	typ GO termu z pohledu databáze GO
acc	char(255)	identifikátor GO termu ve formátu GO:xxxxxxx
is_obs	integer	hodnota 1 pokud je term v aktuální struktuře grafu GO termů již nepoužíván jinak 0
is_root	integer	hodnota 1 pokud je GO term nejvyšším vrcholem ve struktuře grafu GO termů
is_relat	integer	hodnota 1 pokud je GO term ve vztahu s jiným GO termem v tabulce term2term

Tabulka 15: Detailní popis databázové tabulky term2term.

Název sloupce	**Formát hodnoty**	**Význam sloupce**
id	integer	identifikátor záznamu v tabulce
relationship_type_id	integer	druh vztahu mezi term1_id a term2_id z databáze GO
term1_id	integer	identifikátor rodičovského uzlu v grafové struktuře (tabulka term sloupec id)
term2_id	integer	identifikátor uzlu potomka v grafové struktuře (tabulka term sloupec id)
complete	integer	hodnota udávající zda je záznam v tabulce kompletní (údaj za GO databáze)

Literatura

[1] Alberts B. Základy buněčné biologie: úvod do molekulární biologie buňky. Ústí nad Labem: Espero Publishing, 1998. ISBN 8090290604.

[2] Anders S., Huber W. (2010): Differential expression analysis for sequence count data. BioMed Central Genome Biology. 11:R106.

[3] Anders S., Pyl P. T., Huber W. (2014): HTSeq-a python framework to work with high- throughput sequencing data. Bioinformatics. 31, 166 -169.

[4] Camacho C., Coulouris G., Avagyan V., Ma N., Papadopulos J., Bealer K., Madden T.L. (2009): BLAST plus: architecture and applications. BMC Bioinformatics, 10, 421.

[5] Camacho C., Madden T., Ma N., Tao T., Agarwala R., Morgulis A. (2013): BLAST Command line applications user manual. NCBI [online]. [cit. 2016-06-01]. Dostupné z: http://www.ncbi.nlm.nih.gov/books/NBK1763/

[6] Clement M. L., Snell Q., Clement M. J., Hollenhorst P. C., Purwar J., Graves B. J., Cairns B. R., Johnson W. E. (2010): The GNUMAP algorithm: unbiased probabilistic mapping of oligonucleotides from next-generation sequencing. Bioinformatics, 26, 38–45.

[7] Conesa A., Götz S., García-Gómez J. M., Terol J., Talón M., Robles M. (2005): Blast2GO: a univerzal tool for annotation, vizualization and analysis in functional genomics research. Bioinformatics, 21, 3674–3676.

[8] Conesa A., Götz S. (2008): Blast2GO: A comprehensive suite for functional analysis in plant genomics. International journal of plant genomics, ID:619832.

[9] CPAN: Komprehensive Perl Archive Network [online]. [cit. 2016-07-11]. Dostupné z: <http://www.cpan.org/>

[10] Diehl A. D., Lee J. A., Scheuermann R. H., Blake J. A. (2007): Ontology development for biological systems: immunology. Bioinformatics, 23, 913–915.

[11] Duan J., Xia Ch., Zhao G., Jia J., Kong X. (2012): Optimizing de novo common wheat transcriptome assembly using short-read RNA-Seq data. Genomics, 13, 392–404.

[12] Fonseca N. A., Rung J., Brazma A., Marioni J. C. (2012): Tools for mapping hight- throughput sequencing data. Bioinformatics, 28, 3169–3177.

[13] Gene Ontology: Gene Ontology Consortium [online]. [cit. 2016-07-11]. Dostupné z: <http://geneontology.org/>

[14] Grabherr M. G., Haas B. J., Yassour M., Levin J. Z., Thompson D. A., Amit I., Adiconis X., Fan L., Raychowdhury R., Zeng Q., *et al.* (2011): Full-length transcriptome assembly from RNA-Seq data without a reference genome. Nature Biotechnology, 29, 644–652.

[15] Homer N., Merriman B., Nelson S. F. (2009): BFAST: An alignment tool for large scale genome resequencing. Plos One. 4, e7767.

[16] Jones P., Binns D., Chang H.-Y., Fraser M., Li W., McAnulla C., McWilliam H., Maslen J., Mitchell A., Nuka G., Pesseat S., Quinn A.F., Sangrador-Vegas A., Scheremetjew M., Yong S.-Y., Lopez R., Hunter S. (2014). InterProScan 5: genome-scale protein function classification. Bioinformatics, 30(9), 1236-1240.

[17] Lang K. S., Danzeisen J. L., Xu W., Johnson T. J. (2012): Transcriptome mapping of pAR060302, a bla CMY-2 -positive broad-host-range IncA/C plasmid. Applied and environmental microbiology, 78, 3379–3386.

[18] Lin B., Zhang L. F., Chen X. (2014): LFCseq: a nonparametric approach for differential epression analysis of RNA-seq data. BioMed Central Genomics. 15:S7.

[19] Lipman D.J., Pearson W.R. (1985): Rapid and sensitive protein similarity searches. Science, 227, 1435 – 1441.

[20] Love M.I., Huber W., Anders S. (2014): Moderated estimation of fold change and dispersion for RNA-seq data with DESeq2. Genome Biology, 15, 550

[21] Martin J. A., Wang Z. (2011): Next-generation transcriptome assembly. Genetics, 12, 671–682.

[22] Maxam A. M., Gilbert W. (1977): A new method for sequencing DNA. The Proceedings of the National Academy of Sciences, 74, 560–564.

[23] McGinnis S., Madden T.L. (2004): BLAST: at the core of a powerful and diverse set of sequence analysis tools. Nucleic Acids Research, 32, W20-W25.

[24] Metzker M. L. (2009): Sequencing technologies – the next generation. Nature Reviews Genetics. 11, 31 - 46.

[25] Mitchell A., Chang H.-Y.,Daugherty L., Fraser M., Hunter S., Lopez R., McAnulla C., McMenamin C., Nuka G., Pesseat S., Sangrador-Vegas A., Scheremetjew M., Rato C., Yong S.-Y., Bateman A., Punta M., Attwood T.K., Sigrist Ch.J.A., Redaschi N., Rivoire C., Xenarios I., Kahn D., Guyot D., Bork P., Letunic I., Gough J., Oates M., Haft D., Huang H., Natale D.A., Wu C.H., Orengo Ch., Sillitoe I., Mi H., Thomas P.D., Finn R.D. (2015). The InterPro protein families database: the classification resource after 15 years. Nucleic Acids Research, 43, D213-D221.

[26] Mortazavi A., Williams B. A., McCue K., Schaeffer L., Wold B. (2008): Mapping and quantifying mammalian transcriptomes by RNA-Seq. Nature Methods. 5, 621 - 628.

[27] MySQL [online]. [cit. 2016-07-11]. Dostupné z: <http://www.mysql.com/>

[28] NCBI: Reading frames. [online]. [cit. 2016-07-11]. Dostupné z: < http://www.ncbi.nlm.nih.gov/Class/MLACourse/Modules/MolBioReview/readingframes.html

[29] Nirenberg M. (2004) Historical review: Deciphering the genetic code – a personal account. Trends Biochem Sci , 29(1), 46-54.

[30] Oshlack A., Robinson M. D., Young M. D. (2010): From RNA-seq reads to differential expression results. BioMed Central Genome Biology. 11:220.

[31] Perl: The Perl Programing Language. [online]. [cit. 2016-07-11]. Dostupné z: <https://www.perl.org/>

[32] Quinlan A. R., Hall I. M. (2010): BEDTools: a flexible suite of utilities for comparing genomic features. Bioinformatics. 26, 841 - 842.

[33] Robertson G., Schein J., Chiu R., Corbett R., Field M., Jackman S. D., Mungall K., Lee S., Okada H. M., Qian J. Q., *et al.* (2010): De novo assembly and analysis of RNA-seq data. Nature Methods, 7, 909–912.

[34] Robinson M. D., McCarthy D. J., Smyth G. K. (2010): edgeR: a Bioconductor package for differential expression analysis of digital gene expression data. Bioinformatics. 26, 139 - 140.

[35] Ronaghi M., Karamohamed S., Pettersson B., Uhlén M., Nyrén P. (1996): Real-time DNA sequencing using detection of pyrophosphate release. Analytical Biochemistry, 242, 84–89.

[36] Sanger F., Nicklen S., Coulson A. R. (1977): DNA sequencing with chain-terminating inhibitors. The Proceedings of the National Academy of Sciences, 74, 5463–5467.

[37] Schendure J., Ji H. (2008): Next-generation DNA sequencing. Nature Biotechnology, 26, 1135–1145.

[38] Schulz M. H., Zerbino D. R., Vingron M., Birney E. (2012): Oases: robust de novo RNA-seq assembly across the dynamic range of expression levels. Bioinformatics, 28, 1086–1092.

[39] Seyednasrollah F., Laiho A., Elo L. L. (2015): Comparison of software packages for detecting differential expression in RNA-seq studies. Briefings in Bioinformatics. 16, 59 - 70.

[40] Smith L. M., Sanders J. Z., Kaiser R. J., Hughes P., Dodd C., Connell C. R., Heiner C., Kent S. B., Hood L. E. (1986): Fluorescence detection in automated DNA sequence analysis. Nature. 321, 674 - 679.

[41] Soneson Ch., Dolorenzi M. (2013): A comparison of methods for differential expression analysis of RNA-seq data. BioMed Central Bioinformatics. 14:91.

[42] Swerdlow H., Gesteland R. (1990): Capillary gel electrophoresis for rapid, high resolution DNA sequencing. Nucleic Acids Research. 18, 1415 - 1419.

[43] Trapnell C., Pachter L., Salzberg S. L. (2009): TopHat: discovering splice junctions with RNA- Seq. Bioinformatics. 25, 1105 - 1111.Trapnell C., Pachter L., Salzberg S. L. (2009): TopHat: discovering splice junctions with RNA- Seq. Bioinformatics. 25, 1105 - 1111.

[44] Trapnell C., Williams B. A., Pertea G., Mortazavi A., Kwan G., van Baren M. J., Salzberg S. L., Wold B. J., Pachter L. (2010): Transcript assembly and quantification by RNA-Seq reveals unannotated transcripts and isoform switching during cell differentiation. Nature Biotechnology. 28, 511 - 515.

[45] Tucker T., Marra M., Friedman J. M. (2009): Massively parallel sequencing: The next big thing in genetic medicine. The American Journal of Human Genetics, 85, 142– 154.

[46] Turcatti G., Romieu A., Fedurco M., Tairi A. P. (2008): A new class of cleavable fluorescent nucleotides: synthesis and optimization as reversible terminators for DNA sequencing by synthesis. Nucleic Acids Research. 36, e25.

[47] UBUNTU [online]. [cit. 2016-07-11]. Dostupné z: <http://www.ubuntu.com/>

[48] Wang Z., Gerstein M., Snyder M. (2009): RNA-Seq: a revolutionary tool for transcriptomics. Nature Reviews Genetics. 10, 57 - 63.

[49] Wit P. D., Pespeni M. H., Ladner J.T., Barshis D.J., Seneca F., Jaris H., Therkildsen N.O., Morikawa M., Palumbi S.R. (2012): The simple fools guide to population genomics via RNA-Seq: an introduction to hight-throughput sequencing data analysis. Molecular Ecology Resources, 12, 1058–1067.

[50] Wu T. D., Nacu S. (2010): Fast and SNP-tolerant detection of complex variants and splicing in short reads. Bioinformatics, 26, 873–881.