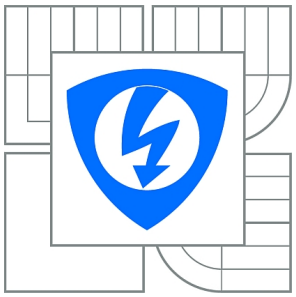
VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

ÚSTAV BIOMEDICÍNSKÉHO INŽENÝRSTVÍ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

DEPARTMENT OF BIOMEDICAL ENGINEERING

VYHLEDÁVÁNÍ HOMOLOGNÍCH GENŮ

HOMOLOGOUS GENE FINDING

DIPLOMOVÁ PRÁCE

MASTER'S THESIS

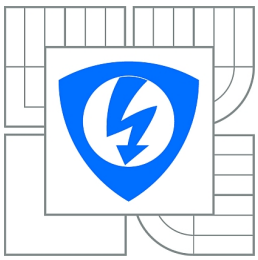
AUTOR PRÁCE — Bc. VERONIKA LÝČKOVÁ

AUTHOR

VEDOUCÍ PRÁCE — Ing. VLADIMÍRA KUBICOVÁ

SUPERVISOR

BRNO 2014



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky a komunikačních technologií

Ústav biomedicínského inženýrství

Diplomová práce

magisterský navazující studijní obor
Biomedicínské a ekologické inženýrství

Studentka: Bc. Veronika Lýčková ***ID:*** 141923
Ročník: 2 ***Akademický rok:*** 2013/2014

NÁZEV TÉMATU:

Vyhledávání homologních genů

POKYNY PRO VYPRACOVÁNÍ:

1) Proveďte literární rešerši zaměřenou na tématiku homologních genů. Pojednejte o algoritmech na vyhledávání homologních genů (BLAST, PatternHunter) a jejich implementaci do komerčních vyhledavačů. 2) V programovém prostředí Matlab realizujte algoritmus založený na zarovnávání studovaných genů. Pojednejte o výběru skórovací matice. 3) Navrhněte metodu, která bude vyhledávat homologní geny pomocí porovnávání proteinů. 4) Výsledky obou metod vzájemně porovnejte a konfrontujte s výsledky z vybrané databáze homologních genů.

DOPORUČENÁ LITERATURA:

[1] CUI, Xuefeng at al. Homology search for genes. Bioinformatics, vol. 23, pp. 97-103, 2007.
[2] SIERK, Michael L. at al. Improving pairwise sequence alignment accuracy using near-optimal protein sequence alignments. BMC Bioinformatics, vol. 11, 2010.

Termín zadání: 10.2.2014 ***Termín odevzdání:*** 23.5.2014

Vedoucí práce: Ing. Vladimíra Kubicová
Konzultanti diplomové práce:

prof. Ing. Ivo Provazník, Ph.D.
Předseda oborové rady

UPOZORNĚNÍ:

Abstrakt

Tato diplomová práce se zabývá problematikou homologních genů a popisem molekulárně biologických databází, které slouží k jejich vyhledávání a vzájemnému porovnávání. Mezi nejdůležitější instituce, které se zabývají správou dat a jejich analýzou, patří EBI, NCBI a CIB. Pro vyhledávání homologních genů je nejzásadnější NCBI – Národní centrum pro biotechnologické informace. Bližší pozornost je věnována vyhledávacím algoritmům homologních genů jako jsou například blastn a PatternHunter. Praktická část této diplomové práce je pak realizace algoritmu srovnávajícího dvě sekvence a nacházejícího homologní geny, a to jak na základě sekvence nukleotidů, tak aminokyselin. Výsledky z vytvořeného programu budou dále konfrontovány s výsledky z komerčně dostupných programů.

Klíčová slova

Homologní geny, NCBI, BLAST, PatternHunter, lokální zarovnání, globální zarovnání

Abstract

This diploma thesis deals with the description of homologous genes and molecular biological databases that are used for their search and allow comparison. Among the most important institutions that deal with data management and analysis, include the EBI, NCBI and CIB. For searching homologous genes is the most fundamental NCBI - National Center for Biotechnology Information. More attention is paid to the search algorithms of homologous genes such as BLASTN and PatternHunter. The practical part of this thesis is the implementation of the algorithm comparing two sequences and locating homologous genes. The comparison is made on the basis of the nucleotide sequence and amino acid sequence. The results generated from the program will be further compared with results from commercially available programs.

Keywords

Homologous genes, NCBI, BLAST, PatternHunter, local alignment, global alignment

LÝČKOVÁ, V. *Vyhledávání homologních genů*. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií. Ústav biomedicínského inženýrství, 2014. 79 s. Diplomová práce. Vedoucí práce: Ing. Vladimíra Kubicová.

PROHLÁŠENÍ

Prohlašuji, že svou diplomovou práci na téma Vyhledávání homologních genů jsem vypracovala samostatně pod vedením vedoucího diplomové práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené diplomové práce dále prohlašuji, že v souvislosti s vytvořením této diplomové práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a/nebo majetkových a jsem si plně vědom následků porušení ustanovení § 11 a následujících zákona č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon), ve znění pozdějších předpisů, včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č. 40/2009 Sb.

V Brně dne ....

(podpis autora)

Poděkování

Na tomto místě bych ráda poděkovala Ing. Vladimíře Kubicové za odborné vedení mé diplomové práce. Dále děkuji panu Mgr. Zdeňku Bonaventurovi, Ph.D. za cenné rady a ochotu pomoci.

V neposlední řadě chci poděkovat svým rodičům a svému partnerovi za podporu a láskyplné zázemí, bez něhož by vytvoření této práce bylo mnohem obtížnější.

Obsah

Úvod

Bioinformatika jako poměrně mladá vědní disciplína, která se pohybuje na hranici informačních technologií, matematiky a biologie, přičemž skloubení těchto znalostí uvádí do praxe v podobě vyhledávání, uchovávání, modelování a zpracování biologických dat. Společně s pokrokem technologií se zvyšuje množství biologických dat, a tedy také vzrůstá potřeba vyvíjení nových informačních systémů zpracovávajících tato data. Bioinformatické poznatky jsou pak dále využitelné v celé řadě oborů – farmakologie, medicína a samozřejmě biologie se všemi svými odvětvími.

Oblastí biologie, která využívá bioinformatiku takřka nejvíce ze všech, je molekulární biologie a genetika. Zde mezi základní témata patří sekvence nukleových kyselin a proteinů a jejich následné srovnávání, vyhledávání genů, tvorba genových map či organizace celého genomu. Cílem bioinformatiky je objasnění informačního obsahu, který nesou biomakromolekuly, a vztah této bioinformace k vývoji a funkci u živých organismů. Bioinformatika také poskytuje důležitá data pro mnohé evoluční studie.

Tato diplomová práce se zabývá problematikou homologií, konkrétně jednotlivými algoritmy pro vyhledávání homologních genů/proteinů, přibližuje jednotlivé molekulárně biologické databáze, v nichž je možné homologie vyhledávat, a také programy a editory, které je zpracovávají. V praktické části je poté vytvořen algoritmus v programu Matlab, který je schopen vyhledávat homologní úseky z nukleotidových/aminokyselinových sekvencí.

1 Obecný biologický úvod

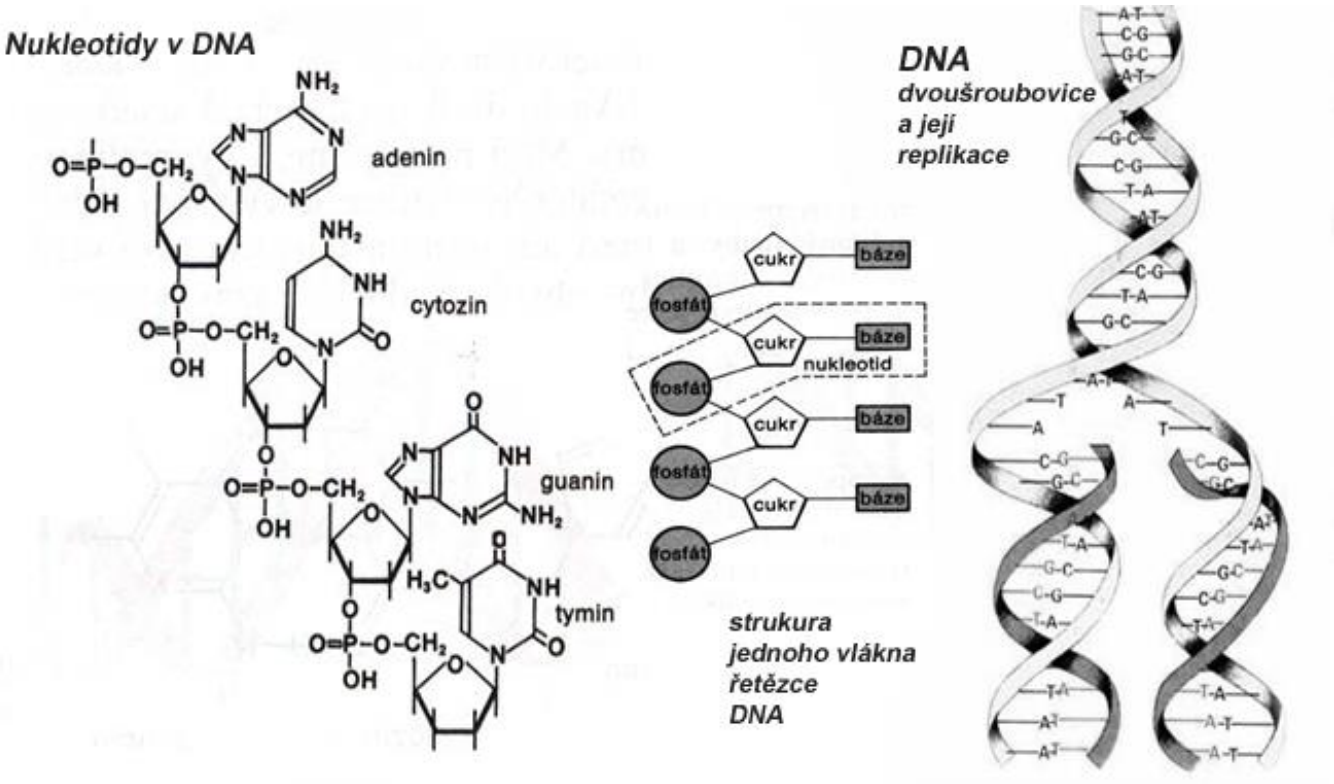
1.1 Nukleové kyseliny

Genetická informace každého živého organismu (řadíme sem i viry, byť jejich řazení mezi živé organismy je diskutabilní) je uložena v nukleové kyselině – DNA či RNA. Struktura nukleové kyseliny byla objasněna v roce 1953 Watsonem, Wilkinsem a Crickem, za což obdrželi v roce 1962 Nobelovu cenu za fyziologii a lékařství [9].

Nukleové kyseliny jsou tvořeny dvěma komplementárními vlákny polynukleotidových řetězců, které jsou navzájem antiparalelní – jejich konce míří opačnými směry. Každý řetězec je tvořen z nukleotidů a jednotlivé nukleotidy jsou tvořeny příslušným sacharidem – ribonukleózou (RNA) nebo deoxyribonukleózou (DNA), fosfátovou skupinou (zbytek po kyselině ortofosforečné) a heterocyklickou bází – purinovou (adeninem, guaninem) či pyrimidinovou (thyminem, cytosinem nebo uracilem). Jednotlivé báze jsou navzájem komplementární a díky této komplementaritě dochází ke spojení obou řetězců. V DNA se váže adenin s thyminem, a to dvojnou vazbou, a cytosin a guaninem vazbou trojnou. V RNA je situace stejná s tím rozdílem, že se váže adenin s uracilem. Vazby mezi bázemi jsou vodíkové. Vazba mezi bází a sacharidem je N-glykosidická. Fosfát a příslušný sacharid tvoří pomocí esterové vazby tzv. cukr-fosfátovou kostru nukleové kyseliny. Struktura nukleové kyseliny je znázorněna na obrázku 1.

DNA, což je nukleová kyselina nacházející se v těle člověka, pak zaujímá prostorové uspořádání jako pravotočivá dvoušroubovice (toto prostorové uspořádání převládá, ale není jediné možné). U eukaryotických organismů je v převážné míře uložena v membránou ohraničeném jádře, ale část se jí nachází také na plasmidech. U prokaryotických organismů nalezneme nukleovou kyselinu převážně volně se pohybující v cytoplasmě v podobě tzv. nukleoidu.

V bioinformatice se primární sekvence makromolekul zapisuje jako takzvaná sekvenční data [13]. Pro DNA a RNA platí pravidlo zápisu sekvenčních dat od 5‘ konce ke 3‘ konci a používají se jednopísmenné IUPAC kódy. IUPAC je zkratkou pro Mezinárodní unii pro čistou a užitou chemie (The International Union of Pure and Applied Chemistry). Přehled značení pro sekvence nukleotidů je uveden v tabulce 1.



Obrázek 1: Struktura nukleotidového řetězce nukleové kyseliny (primární struktura) a pravotočivá šroubovice DNA (sekundární struktura). Převzato z [7].

Tabulka 1: IUPAC kódy pro zápis sekvencí nukleotidů. Převzato z [13].

DNA, RNA	
kód	**Báze**
A	Adenin
C	Cytosin
G	Guanin
T	Thymin
U	Uracil
R	A, G *(purine)*
Y	C, T *(pyrimidine)*
S	G, C *(strong)*
W	A, T *(weak)*
K	G, T *(keto)*
M	A, C *(amino)*
B	C, G, T *(not A)*
D	A, G, T *(not C)*
H	A, C, T *(not G)*
V	A, C, G *(not T)*
N	cokoli *(any)*
. nebo ,	Mezera

1.2 Proteiny

Proteiny jsou základními stavebními a funkčnímu jednotkami všech organismů [27]. V organismech zastávají celou řadu životně nezbytných funkcí. Patří mezi ně například:

- detekují a přenáší signály a informace
- regulují činnost buněk
- transdukují energii
- katalyzují chemické reakce
- jsou zodpovědné za transport látek
- jsou zodpovědné za strukturu organismu
- …

Sekvence aminokyselin se jako sekvenční data zapisují od N – konce k C – konci. Přehled značení pro sekvence aminokyselin je uveden v tabulce 2.

Tabulka 2: IUPAC kódy pro zápis sekvencí aminokyselin. Převzato z [13].

Protein		
kód	**zkratka**	**Aminokyselina**
A	Ala	Alanin
C	Cys	Cystein
D	Asp	Aspartát
E	Glu	Glutamát
F	Phe	Fenylalanin
G	Gly	Glycin
H	His	Histidin
I	Ile	Izoleucin
K	Lys	Lysin
L	Leu	Leucin
M	Met	Methionin
N	Asn	Asparagin
P	Pro	Prolin
Q	Gln	Glutamin
R	Arg	Arginin
S	Ser	Serin
T	Thr	Threonin
V	Val	Valin
W	Trp	Tryptofan
Y	Tyr	Tyrosin
X	Xxx	Cokoli
B	Asx	aspartát nebo aspargin
Z	Glx	glutamát nebo glutamin

1.3 Geny, jejich dělení a využití

Geny jsou základní jednotky genetické informace. Soubor genů v celém organismu vytváří tzv. genotyp [1]. V souvislosti s genotypem je třeba zmínit také pojem fenotyp, což je vnější projev genotypu v organismu. Fenotyp nám tedy určuje vnější podobu organismu. V organismu nalezneme různé druhy genů, avšak nejdůležitější význam mají geny strukturní. Velmi významné jsou pak také geny regulační, které regulují genetické procesy v organismu.

Strukturní geny jsou z pohledu molekulární biologie tvořeny sekvencí nukleotidů. Nukleotidová sekvence je molekulárními procesy transkribována a translatována a nese informaci o konečném produktu genů – ve většině případů se jedná o informaci pro sestavení polypeptidového řetězce, ale může se jednat také např. o různé druhy RNA. Tuto cestu přenosu informace popisuje centrální dogma molekulární biologie, které hovoří o možnosti přepisu informace z nukleové kyseliny do RNA, z níž je dále překládána do struktury bílkovin, avšak reverzně tento proces možný není.

V případě eukaryot jsou geny tvořeny exony (kódující oblasti) a introny (nekódující oblasti). Po transkripci je třeba provést tzv. sestřih, kdy jsou odstraněny nekódující oblasti a do primární struktury bílkovin jsou dále překládány pouze oblasti kódující. V případě prokaryot je situace výrazně jednodušší, jelikož jejich geny jsou tvořeny pouze exony, a tedy sestřih není nutný. Jedinou výjimku ve světě mikroorganismů tvoří skupina Archea a bakteriofágy, které introny obsahují.

Geny lze dělit do několika základních skupin [2]:

1) Homology – homologní geny jsou takové, které se vyvinuly z jednoho společného genu. Tento společný gen se označuje jako ancestrální. Homologní geny pak byly dále rozděleny do dvou podskupin:
   a) Ortology – ortologní geny vznikly speciací (jedná se o biologický evoluční pochod, při němž se vývojové linie štěpí, čímž dochází ke vzniku nových taxonomických druhů; název je odvozen od latinského *species* – druh).
   b) Paralogy – paralogní geny vznikly duplikací ancestrálního genu. Speciální skupinou paralogů je následující skupina:
      - Ohnology – jedná se geny vzniklé duplikací celého genomu [49].
2) Heterology – geny neodvozené z jednoho ancestrálního genu. Svou variabilitu získaly pomocí přenosu genetické informace, a to především metodou mezidruhového horizontálního přenosu.

Další pojem, s nímž je možné se setkat v souvislosti s homologními geny, je pojem xenology. Jedná se o homologní geny vzniklé horizontálním přenosem [50].

V souvislosti s proteiny pak ještě hovoříme o tzv. analozích. Analogy jsou proteiny, které mají stejnou funkci, např. katalyzují stejnou reakci, ale nejsou si strukturálně příbuzné.

Sekvence kodonů, tj. trojice nukleotidů kódující aminokyseliny, jsou evolučně konzervované. Na základě toho můžeme určit příbuznost mezi druhy pomocí srovnání nukleotidových sekvencí. Obdobně lze srovnávat také proteinové sekvence.

1.4 Homologní sekvence

S pojmem homologie není možné pracovat jako s veličinou, která má nějaký rozsah. Je nutno brát ji absolutně. Tedy není možné říci, že dvě sekvence jsou navzájem homologní z 50%. Lze říci pouze to, zdali jsou si podobné či nikoliv. Vzhledem k tomuto faktu je nezbytné stanovit si práh, od nějž již lze považovat nalezenou podobnost sekvencí za skutečnou homologii. Ve většině komerčně dostupných programů je tímto prahem hodnota 35% (jde například o programy HOGENOM, PSAT – Prokaryotic Sequence Analysis Tool) [51]. Pokud je tedy sekvenční identita nad 35%, jedná se pravděpodobně o homologní sekvenci. Je sekvenční identita v rozmezí 20 – 35%, sekvence mohou být homologní, ale nemusí (jde o tzv. „twilight zone“). Pokud je sekvenční identita pod 20%, s nejvyšší pravděpodobností se nejedná o homology. V komerčně dostupných programech jsou pak dále zhodnoceny některé další statistické parametry, které ukazují na průkaznost či naopak na dílo náhody nalezení shody (blíže viz kapitola 4.1.1).

Pokud je vznesena otázka, zdali je lepší určovat homologie ze sekvence nukleotidové či proteinové, tak odpověď je víc než logická – spolehlivější je určovat homologie ze sekvence aminokyselin, jelikož možné nukleotidy jsou čtyři (A, T, G, C) a pravděpodobnost shody mezi čtyřmi písmeny je výrazně vyšší než mezi dvaceti (což je počet esenciálních aminokyselin). Pokud už tedy dojde v sekvenci aminokyselin ke shodě v pozici, je tato shoda výrazně průkaznější než v případě nukleotidové sekvence.

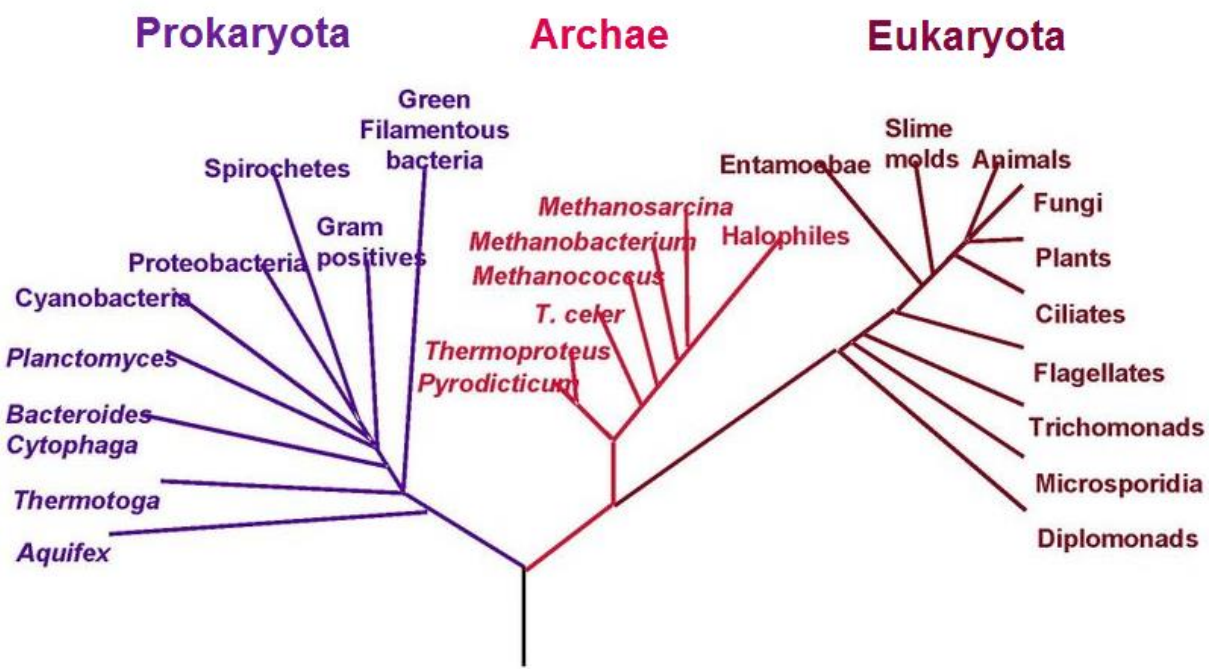
Názornou ukázkou anatomické homologie z říše savců je například evoluce netopýřích křídel a lidských horních končetin [47]. Tento znak, jak fenotypový, tak genotypový, nese základ v jednom jediném ancestrálním předkovi – struktury se vyvinuly ze stejného typu tkáně během embryonálního vývoje. Situací, kde by bylo možno se mylně domnívat, že se jedná o homologii, jsou křídla netopýra a křídla ptáka. Tento znak se vyvinul zcela nezávisle, každý v jiné linii z jiného předka, byť fenotypový projev je velice podobný.

1.4.1 Význam a detekce homologií v biologii

Nalezení shodných nukleotidových či proteinových sekvencí má velké uplatnění v oboru nazývajícím se fylogenetika. Jedná se o vědu, která se zabývá vývojem druhů, dále vývojem mnoha fyziologických či patologických znaků organismů nebo také slouží k detekci genových mutací [3].

Fylogenetika je tedy věda, která na základě poznatků o homologních sekvencích sestavuje genetické stromy, v nichž je znázorněna podobnost mezi jednotlivými druhy a také cesta k jejich společnému předku. Existuje fylogenetický strom zakořeněný (zde je znázorněn

společný předek všech zkoumaných taxonů) a nezakořeněný (není znázorněn společný předek). Dříve se fylogenetické stromy tvořily pouze na základě morfologické podobnosti, což se později ukázalo jako znak nevhodný a přikročilo se k již zmíněným molekulárně – biologickým metodám. Tvorba fylogenetického stromu je závislá na celé řadě poměrně náročných matematických výpočtů, je třeba správně stanovit délky větve, a také tzv. bootstrap – stanovení spolehlivosti větví. Příklad fylogenetického stromu je znázorněn na obrázku 2.



Obrázek 2: Fylogenetický strom říší Prokaryota, Archae, Eukaryota. Převzato a upraveno z [48]. Pozn.: Zde je nutno zdůraznit, že obzvláště větve prokaryot a archae neustále procházejí značným vývojem, jelikož bylo přistoupeno k novým metodám konstrukce těchto stromů (analýza na základě 16S rRNA), a tedy dělení do jednotlivých kmenů, tříd a nižších taxonů podléhá neustálému vývoji.

Metod, jak určit podobnost nukleové kyseliny či proteinu, je celá řada. Patří sem například DNA fingerprinting, gelová elektroforéza a v neposlední řadě také stále se rozšiřující bioinformatické nástroje, které pomocí tzv. alignmentu = zarovnání sekvencí jsou schopny poskytnout velmi přesné výsledky při analýze podobnosti sekvencí. Pomocí zarovnání dojde k přiřazení poziční homologie.

Z biologického odvětví pak zcela přesnou metodu určení sekvence nukleotidů zastupuje tzv. sekvenování – jedná se o poměrně náročnou biochemickou metodu. V současné době se v praxi provádí spíše sekvenování nukleotidů, jelikož sekvenování proteinů je časově i finančně podstatně náročnější.

Poté, co máme zarovnanou či osekvenovanou sekvenci nukleotidů/proteinů, je třeba pomocí dalšího programu a algoritmů zjistit jejich vzájemnou podobnost.

Sekvence nukleotidů a proteinů a jejich podobnosti jsou shromažďovány v molekulárně biologických databázích.

2 Molekulárně biologické databáze

V roce 2008 bylo dostupných přibližně 550 bioinformatických databází [11]. Jejich aktuální přehled je každoročně publikován v časopise Nucleic Acid Research. Existují tři nejdůležitější instituce, které se zabývají správou bioinformatických dat: EBI, NCBI a CIB. Každá z těchto tří institucí spravuje nějakou databázi a má svůj vlastní vyhledávací systém, který umožňuje rychlé nalezení požadovaných informací a jejich zpřístupnění v jednotné formě. Tyto vyhledávací systémy jsou vstupem do mnoha integrovaných databází. Všechny databáze spolu také vzájemně spolupracují.

EBI, NCBI a CIB tvoří „velkou trojici" databází, kde běžný uživatel je schopen najít v podstatě veškeré informace, které potřebuje. Existují však pochopitelně databáze specializované například na strukturu proteinu, sekvenci proteinu, genomové databáze, databáze taxonomických podobnost atd. Krom „velké trojici" zde pouze okrajově zmíním dvě nejvýznamnější proteinové databáze.

V září roku 2003 bylo v primárních databázích uloženo $27{,}2 \cdot 10^6$ záznamů, jejichž zdrojem je zhruba 150 000 různých organismů [28]. Je třeba brát v potaz, že objem sekvenčních data narůstá nesmírně rychle, a je tedy nezbytné dodržovat určitá pravidla zápisu a následné údržby.

2.1 EBI

EBI znamená Evropský institut pro bioinformatiku (Energy Biosciences Institute). Jeho sídlo je v Hinxtonu ve velké Británii. EBI umožňuje přístup do databáze EMBL-Bank, která byla zřízena jako vůbec první databanka nukleotidových sekvencí již v roce 1980 [11]. Zřizovatelem této databáze byla Evropská molekulárně biologická laboratoř (odtud tedy i samotný název databáze EMBL = European Molecular Biology Laboratory). Databáze je dostupná na internetové adrese http://www.ebi.ac.uk/ena/. Vyhledávací systém vyvinutý v EBI nese název SRS (Sequence Retrieval System). Do EMBL-Bank lze vložit nové sekvence pomocí www formuláře Webin http://www.ebi.ac.uk/embl/Submission/webin.html nebo pomocí programu Sequin. Seuqin je vhodný především pro vkládání delších sekvencí, typicky například celých genomů, a je možné jej stáhnout zdarma na této adrese: http://www.ncbi.nlm.nih.gov/Sequin/download/seq_download.html.

K tomu, aby uživatel mohl vložit do databáze novou sekvenci, je nezbytné, aby vyplnil několik informací: kontaktní údaje o místě (univerzita, laboratoř atd.), kde proběhl výzkum dané sekvence, informaci o datu zveřejnění, o publikacích relevantních k této sekvenci nebo například popis zdroje sekvence (zdali se jedná o DNA, RNA, jaký typ dané nukleové kyseliny atd.).

2.2 NCBI

NCBI je Národní institut pro biotechnologické informace (National Center for Biotechnology Information) [10]. Jeho původ je v USA a nejprve byl součástí Národní lékařské knihovny (NLM). Nyní NCBI sídlí v Bethesdě ve státě Maryland. Databáze, kterou spravuje NCBI nese název GenBank a vytvořena byla v roce 1992. Dalšími databázemi spadajícími pod NCBI je například PubMed, což je databáze pro vyhledávání bibliografií. Přístupná je na internetové adrese http://www.ncbi.nlm.nih.gov/. Vyhledávací systém vyvinutý v NCBI se jmenuje Entrez. Do GenBank lze vložit nové sekvence pomocí www formuláře Banklt http://www.ncbi.nlm.nih.gov/BankIt/ nebo pomocí programu Sequin.

„Hlavou NCBI“ je David Lipman, který mimojiné stojí také za zrodem BLASTu a mnoha dalších významných bioinformatických projektů.

2.3 CIB

CIB – Centrum pro informační biologii (Center for Information Biology) je částí Národního genetického institutu (NIG) v Mishimě v Japonsku. CIB spravuje databázi DDBJ (DNA Data Bank of Japan), která shromažďuje především data z japonských výzkumů, a to již od roku 1984. Databáze je dostupná na internetové adrese http://www.ddbj.nig.ac.jp/. Na Univerzitě Kyoto byl vyvinut vyhledávací systém DBGET/Link DB (Integrated Database Retrieval System). Do DDBJ lze vložit nové sekvence pomocí www formuláře Sakura http://sakura.ddbj.nig.ac.jp/ nebo pomocí programu Sequin.

2.4 Proteinové databáze

Při vyhledávání sekvencí proteinů je samozřejmě možné použít jednu z databází z již zmíněné „velké trojice“ (EBI, NCBI, CIB), avšak jejich drobnou nevýhodou je, že se jedná o tzv. nemoderované databáze, což znamená, že do nich může přispívat v podstatě kdokoliv (samozřejmě pokud vyplní náležité informace o sekvenci) [13]. Správce nemoderované databáze však nekontroluje, zdali je vkladatel sekvence dostatečně kvalifikovaný či kompetentní. Moderované databáze kladou na vkládání sekvencí mnohem větší požadavky – například zveřejnění této sekvence ve vědeckém časopise i s recenzí apod.

Mezi takovéto moderované databáze patří například významná proteinová databáze Swiss-Prot, což je databáze sekvencí nebo databáze strukturní jako je Protein Data Bank (RCSB PDB).

2.4.1 Swiss-Prot

Swiss-Prot je databáze spadající pod UniProtKB (přístup z http://www.uniprot.org/, zkratka Universal Protein Resource Knowledgebase), což je centrální úložiště proteinových sekvencí a funkčních informací o proteinech [29]. Druhou databází spadající pod UniProtKB je TrEMBL (Translated European Molecular Biology Laboratory). Od Swiss-Prot se liší počtem záznamů a kvalitou anotací – TrEMBL je provádí automaticky, kdežto Swiss-Prot manuálně. V TrEMBL se v podstatě nachází překlady nukleotidových sekvencí z EMBL, které dosud nebyly integrovány do Swiss-Prot.

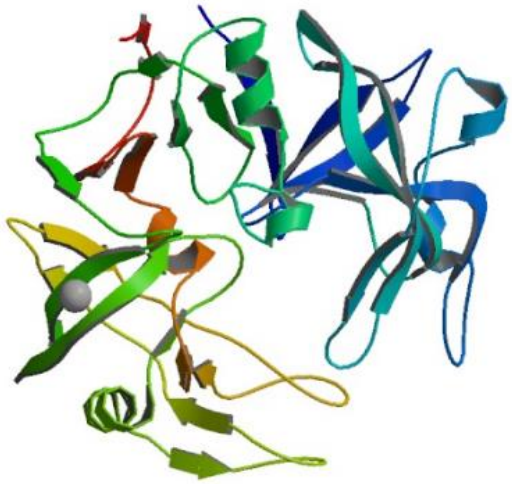
Na tvorbě této databáze se velmi významně podílí Swiss Institute od Bioinformatics a Protein Informatin Resource. Tento fakt, mimo jiné, přispívá k tomu, že se jedná o databázi proteinových sekvencí, která se vyznačuje velkou spolehlivostí z důvodu experimentálního ověření a kvalitních a bohatých anotací hovořících například o biologické ontologii, klasifikaci, informacích o funkci proteinu, post-translačních modifikacích apod. Informace k proteinové sekvenci mohou být postupně doplňovány, avšak pouze po splnění přísných podmínek kvality.

2.4.2 Protein Data Bank (PDB)

Tato databáze je centrálním světovým úložištěm proteinových struktur (mimo to obsahuje struktury dalších makromolekul, jako jsou nukleové kyseliny) a je dostupná z internetové adresy http://pdb.rcsb.org/ [30]. Provozována je Výzkumnou spoluprací pro strukturní bioinformatiku (Research Collaboratoryfor Structural Bioinformatics).

Struktury uložené v této databázi byly získány buďto pomocí NMR (Nucler Magnetic Resonance – nukleární magnetická rezonance), rentgenovou krystalografií nebo elektronovou mikroskopií. K vložení struktury do RCSB PDB je nezbytná publikace této struktury ve vědeckém časopise.

Výstupem z PDB jsou informace o velikosti, struktuře proteinu a také model 3D struktury, který je uživatelsky velmi atraktivní. Příklad je na obrázku 3.



Obrázek 3: 3D struktura sekretované aspartanové proteinázy (Sap) 3 z *Candida albicans*. Zdroj: http://pdb.rcsb.org

2.5 Formáty biologických dat

Formátů, v nichž pracovat s biologickými daty, je celá řada. Nejjednodušší je prostý nukleotidový zápis bez jakýchkoliv úprav (tzv. raw data – surová) [10]. Tato varianta však není optimální pro další zpracování a vyhledávání. Z tohoto důvodu byly vytvořeny standardizované formáty, v nichž lze s daty pracovat. Takto vytvořených formátů existuje nepřeberné množství, proto zde uvedu jenom ty nejdůležitější.

Je nutno dodat, že pro každý záznam vložený do databáze je určen unikátní přístupový kód (accesion), který je složen z čísel a písmen [21]. Původní verze přístupového kódu se skládala z jednoho velkého písmene a pěti čísel, nové záznamy obsahují dvě velká písmena a šest čísel. Tento přístupový kód zůstává po celou dobu existence záznamu v databázi nezměněn a je možné podle něj záznam vyhledat.

2.5.1 FASTA

FASTA formát je nejjednodušší a nejčastěji používaný formát (krom již zmíněného neupraveného zápisu). První řádek tohoto formátu začíná znakem „>" a za ním následuje jednoduchý popis sekvence (její název, anotace) [13]. Právě číselný popis sekvence je jejím jedinečným identifikátorem. FASTA formát se neliší pro nukleové kyseliny a proteiny a neobsahuje takřka žádné dodatečné informace o sekvence. Jednotlivé sekvence jsou zapsány po sobě jdoucími písmeny, která vyjadřují jednotlivé báze/aminokyseliny

v nukleotidové/proteinové sekvenci. Mezi jednotlivými znaky sekvence není žádná mezera a ani cizí znak. Tento zápis svou jednoduchostí umožňuje velice snadnou manipulaci a analýzu pomocí textového zpracování a skriptovacích jazyků jako je např. Python.

Příkladem je úryvek ze zápisu genové sekvence lidského chromozomu X ve formátu FASTA (zdroj: http://www.ncbi.nlm.nih.gov/):

```
>gi|82583714|emb|CT009680.1| Human chromosome X complete sequence
CTAACCCTAACCCTAACCCTAACCCTAACCCTAACCCTCTGAAAGTGGACCTATCAGCAGGATGTGGGTG
GGAGCAGATTAGAGAATAAAAGCAGACTGCCTGAGCCAGCAGTGGCAACCCAATGGGGTCCCTTTCCATA
CTGTGGAAGCTTCGTTCTTTCACTCTTTGCAATAAATCTTGCTATTGCTCACTCTTTGGGTCCACACTGC
CTTTATGAGCTGTGACACTCACCGCAAAGGTCTGCAGCTTCACTCCTGAGCCAGTGAGACCACAACCCCA
```

Dalším příkladem je úryvek ze zápisu proteinové sekvence enzymu polymerázy viru hepatitidy B (zdroj: http://www.ncbi.nlm.nih.gov/):

```
>gi|431983379|gb|AGA95798.1| polymerase [Hepatitis B virus]
MPLSYQHFRKLLLLDEEAGPLEEELPRLADEGLNRRVAEDLNLGNLNVSIPWTHKVGNFTGLYSSTVPCF
NPKWQTPSFPDIHLQEDIVDRCKQFVGPLTVNENRRLKLIMPARFYPNVTKYLPLDKGIKPYYPEHVVNH
YFQTRHYLHTLWKAGILYKRESTRSASFCGSPYSWEQDLQHGRLVFKTSKRHGDKSFCPQSPGILPRSSV
GPCIQSQLRKSRLGPQPAQGQLAGRQQGGSGSIRARVHPSPWGTVGVEPSGSGHTHNCASSSSSCLHQSA
```

2.5.2 EMBL a GenBank

V tomto formátu začíná samotný obsah až od šestého znaku, první dva znaky na řádku pak pojmenovávají identifikátor [12]. Identifikátory mezi sebou jsou odděleny znaky „XX“ a identifikátor je vždy dvoupísmenný záznam pak končí znaky „//“. Dvoupísmenný záznam tedy vždy určuje, co je na konkrétním řádku zaznamenáno. Řádky se pak uveřejňují v takovém pořadí, v jakém byly zapsány (vyjma řádku „XX“).

Formát pro GenBank je velice podobný formátu EMBL, rozdíl je však v tom, že identifikátory u GenBank jsou celá slova.

Jak již bylo zmíněno v kapitole 2.5, vložená sekvence získává přístupový kód. Společně s ním je jí také od roku 1992 po publikování v GenBank uděleno takzvané „GI number“ (GenBank Identifier) [21]. GI number je udělováno všem sekvencím zapsaným přes Entrez, ale také záznamy z DDBJ, Swiss-Prot a další. GI number je reprezentováno pouze celými čísly a slouží k další identifikaci záznamu v databázi. Tak, jako by se dal přístupový kód přirovnat k číslu občanského průkazu, tak GI number k číslu řidičského průkazu.

Příklad zápisu nukleotidové sekvence kosmidu *Streptomyces coelicolor* (zdroj: http://www.ebi.ac.uk/):

```
CC   Cosmid 10H5 lies to the right of 3A7 on the AseI-B genomic restriction
CC   fragment.
XX
FH   Key             Location/Qualifiers
```

```
FH
FT   source          1..4870
FT                   /organism="Streptomyces coelicolor"
FT                   /strain="A3(2)"
FT                   /clone="cosmid 10H5"
FT   CDS             complement(<1..327)
```

Příklad zápisu části nukleotidové sekvence lidského chromozomu X ve formátu GenBank (zdroj: http://www.ncbi.nlm.nih.gov/):

```
misc_feature    <1..>519
                     /note="annotated region of clone"
     gene            255..>519
                     /gene="PPP2R3B"
                     /locus_tag="LL0YNC03-56G10.1-002"
     mRNA            255..>519
                     /gene="PPP2R3B"
                     /locus_tag="LL0YNC03-56G10.1-002"
                     /product="protein phosphatase 2 (formerly 2A),
regulatory
                     subunit B'', beta"
                     /note="match: cDNAs: Em:AF135016.1 Em:AF155098.1
                     Em:BC011180.1;
                     match: ESTs: Em:AI248788.1 Em:AI926984.1 Em:AL577905.1
```

3 Posuzování podobnosti sekvencí

Srovnat mezi sebou dvě (či více) sekvence je jedním z ústředních témat bioinformatických studií. Hlavním cílem takového zarovnávání sekvencí je nalézt shodu bází na téže pozici v porovnávaných sekvencích. Při správném zarovnání sekvencí je nalezena evoluční příbuznost dvou sekvencí a zhodnocena míra této příbuznosti.

Rozdílně se k zarovnávání sekvencí přistupuje v případě, že jsou analyzovány dvě stejně dlouhé či nepříliš odlišné sekvence a odlišný přístup je v případě sekvencí velmi odlišných či nestejně dlouhých. Pokud jsou sekvence podobné a stejně dlouhé, probíhá zarovnání poměrně jednoduše ve dvou krocích [13]. Prvním krokem je tzv. přiřazení (alignment). Přiřazení je proces, při němž se začátek analyzovaných sekvencí umístí pod sebe. Poté je stanoveno tzv. skóre, které vyčísluje míru podobnosti sekvencí. Skóre se vyčísluje pro každou přiřazenou dvojici zvlášť. Hodnota skóre nabývá, při tomto nejjednodušším srovnání sekvencí, dvou možných variant: match = shoda a mismatch = neshoda.

Tato nejjednodušší metoda je ukázána na následujících sekvencích. Cílem je zodpovězení otázky, zdali jsou si podobnější sekvence A a B nebo C a D:

```
A = ATTGCTCTGT
B = ATAGCTCGGT
C = ATTGCACTGTAATGCCATGT
D = ATTGCTCTGAAATGCCCTGT
```

Přiřazení (alignment):

```
A = A T T G C T C T G T
    | |   | | | |   | |
B = A T A G C T C G G T

C = A T T G C A C T G T A A T G C C A T G T
    | | | | |   | | |   | | | | | |   | | |
D = A T T G C T C T G A A A T G C C C T G T
```

Výpočet skóre – shoda pozice je ohodnocena jako 1 a neshoda jako 0 a celkové skóre je počítáno jako podíl součtu všech shod a neshod s celkovým počtem párů:

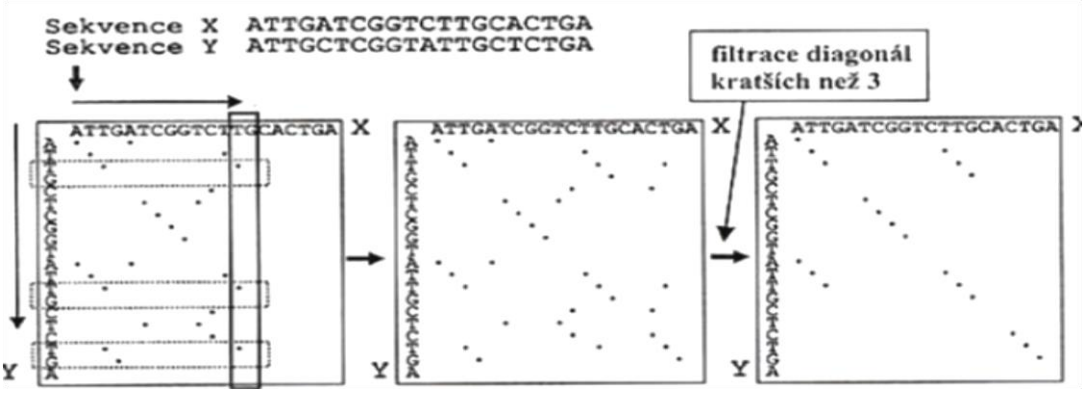
$\text{Skóre}_{AB} = \frac{8 \cdot 1 + 2 \cdot 0}{10} = 0{,}80 \rightarrow 80\%$

$$\text{Skóre}_{CD} = \frac{17 \cdot 1 + 3 \cdot 0}{20} = 0{,}85 \rightarrow 85\%$$

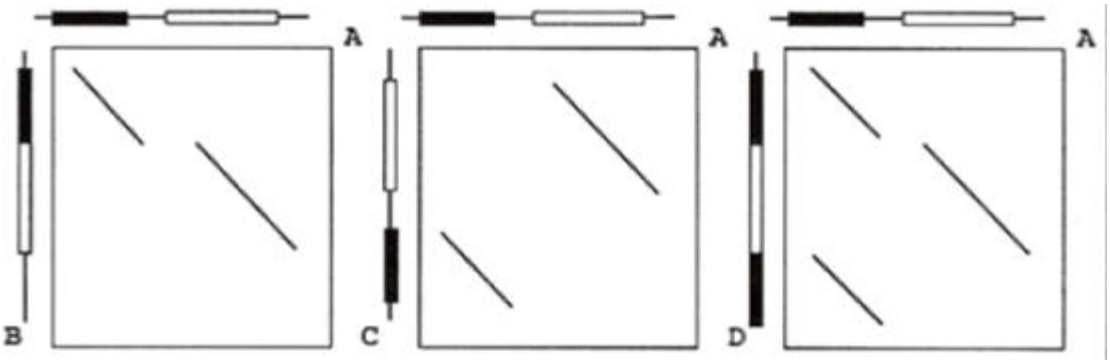
Z tohoto vyplývá, že podobnější jsou si sekvence C a D.

Tato metoda však není aplikovatelná při srovnávání sekvencí, které jsou nestejně dlouhé nebo až příliš odlišné. V těchto případech je pak na výběr ze dvou metod: globální zarovnání a lokální zarovnání [21].

Nejjednodušší metodou, jak zvolit, který způsob zarovnání je výhodnější použít, je sestavení tzv. bodového diagramu (dotplot). Sestavení takového bodového diagramu je poměrně snadný proces. Na osy X a Y jsou vyneseny sekvence, které analyzujeme, a to tak, aby pozice první báze v první sekvenci odpovídala pozici první báze v druhé sekvenci [13]. Dále je zvoleno tzv. „okno“ o vhodné velikosti, které „klouže“ po sekvenci po kroku o velikosti 1. Správně zvolené okno redukuje šum vznikající u příliš dlouhých a nepřehledných sekvencí. Je-li nalezena shoda v bázi na ose X a bázi na ose Y, do grafu je vynesen bod (dot). Tímto způsobem vzniknou diagonály. Dále se pracuje pouze s těmi diagonálami, které jsou delší než předem určená délka. Takto vybrané diagonály vyjadřují podobnost sekvence. Popsaný proces tvorby bodového diagramu je znázorněn na obrázku 4. Nejprve je zkonstruován bodový diagram (obrázek 4 A) – tečka značí shodu pozic sekvencí vynesených na ose X a Y. Z takto vynesených shod je patrno, že podobnost dvou sekvencí se jeví jako diagonála. Dále jsou diagonály, které obsahují méně než 3 body, odfiltrovány a zůstanou pouze delší diagonály. Na základě takto sestaveného bodového diagramu lze rozlišit, jestli jsou sekvence vhodné pro globální či lokální zarovnání (obrázek 4 B). Zde je patrné, že pouze první dvojice sekvencí, tedy A a B, je vhodná pro globální zarovnání. U zbývajících dvou sekvencí (A a C a A a D) se dá předpokládat, že došlo k různým přestavbám genomu, a tedy globální zarovnání není možné a je nutno zarovnat lokálně.



(A)



(B)

Obrázek 4: Konstrukce (A) a vyhodnocení (B) bodového diagramu. Převzato a upraveno z [13].

3.1 Globální a lokální zarovnání jako metody vyhledávání homologních genů

Globální zarovnání bere při každém přiřazení ohled na celou sekvenci nukleotidů. Je při něm možné vnášet do sekvence mezery. Jeho použití je vhodné tehdy, když jsou od sebe sekvence odlišné jen málo. Globální zarovnávání tedy pracuje se všemi vstupními informacemi o sekvencích, a tedy je výpočetně poměrně hodně náročné. Tento typ zarovnání je výhodné použít při srovnávání sekvencí, které si nejsou evolučně příliš vzdáleny. Pokud totiž zvolíme tuto metodu v nesprávném případě, můžeme získat zkreslené výsledky.

Při použití globálního zarovnání je pak třeba při výpočtu finálního skóre podobnosti zohlednit vkládání mezer a započítat je jako penalizace. Globální zarovnání je realizovatelné pomocí algoritmu Needleman-Wunsch.

Lokální zarovnání zarovnává pouze úseky sekvencí, které jsou si podobné, a je ukončeno ve chvíli, kdy narazí na větší rozdíly. Vyznačuje se tím, že do sekvencí nejsou vnášeny mezery. Lokální zarovnání pracuje s algoritmem Smith-Waterman.

3.2 Substituční (skórovací) matice

K ohodnocení každé pozice bez mezer je použita substituční matice. Substituční matice pro zarovnání nukleotidů jsou poměrně jednoduché, pro zarovnání proteinů už výrazně složitější. Nejmenší možná velikost substituční matice je určena počtem základních bází v DNA (tzn. 4 – adenin, thymin, cytosin, guanin), přičemž do tohoto počtu nejsou zahrnuty možnosti, kdy přesně neznáme druh báze v nukleotidu [31]. V sekvenci proteinů je minimální velikost matice určena počtem esenciálních aminokyselin. V současné době je tedy třeba počítat i s aminokyselinami pyrrolizinem a selenocysteinem, které byly k esenciálním aminokyselinám přiřazeny relativně nedávno, a tedy minimální velikost matice je 22×22. Velikosti matice 4 a 22 jsou tedy nejmenší možné a dá se s nimi pracovat v situacích, kdy se předpokládá naprostá znalost dotazované sekvence, tedy u každé pozice báze či aminokyseliny je přesně známo, o jakou se jedná. Matice běžně užívané v komerčních vyhledávačích však počítají s možnými „neznalostmi“ některých bází či aminokyselin (celkový možný počet viz tabulka 1 a 2) a jejich velikost je pak výrazně větší.

Po stanovení velikosti matice je vždy třeba stanovit si ohodnocení shody (match) a neshody (mismatch) pozice báze/aminokyseliny [32]. Substituční matice je symetrická podle diagonály. Ohodnocení shod a neshod v nejběžněji používaných maticích vychází z experimentů, a to především genetických (jedná se například o sledování frekvencí nukletidových/aminokyselinových záměn ve známých homologních sekvencích). Konkrétní hodnoty v matici jsou poté stanoveny na základě empirie, přičemž konvencí je shody ohodnocovat kladně a neshody záporně.

V případě sekvencí aminokyselin je také třeba zhodnotit váhu určitých substitucí. Často se vyskytující substituce, jako je například substituce aspartát – glutamát, je hodnocena vyšším skóre než málo frekventovaná substituce, například aspartát – leucin [31].

3.2.1 Matematický výpočet

V bioinformatice existuje mnoho dat a jevů, u nichž je třeba zohlednit míru náhody, která zasáhla. Nejinak je tomu také při tvorbě substitučních matic.

Při výpočtu matice použité pro srovnání aminokyselin je třeba také zohlednit, že ne všechny substituce mají stejnou váhu – některé mohou být (na základě evolučních experimentů) vzácné a jiné časté. V tomto případě hovoříme o tzv. log odds ratio substituci (LogLR) [52]. LogLR počítá, s jakou pravděpodobností si budou dvě sekvence rovny za předpokladu, že mají společného předka a za předpokladu, že jde o pouhé dílo náhody.

Máme tedy dvě sekvence – A a B, výpočet logLR je následující:

$$\log LR(A,B) log\left(\frac{P(A,B/předek)}{P(A,B/náhoda)}\right) \tag{3.1}$$

Samotná čísla v matici jsou poté sestavena na základě tzv. „log odds“ – měřítko relativní mutovatelnosti („logaritmus poměru šancí). Tvorba log odds je popsána na konkrétním příkladu:

Máme dvě sekvence: ABC a DEF. Při přiložení těchto dvou sekvencí k sobě jsou možné dvě hypotézy, které nastanou:

a) Pravděpodobnost, že dojde k přiložení ABC a DEF při platnosti hypotézy společného předka:

$$P\left(\frac{\text{přiložení}}{spol.\text{předek}}\right) = P\left(\frac{A,D}{spol.\text{předek}}\right) \times P\left(\frac{B,E}{spol.\text{předek}}\right) \times P\left(\frac{C,F}{spol.\text{předek}}\right) \quad (3.2)$$

b) Pravděpodobnost, že dojde k přiložení ABC a DEF při platnosti náhodné hypotézy:

$$P\left(\frac{\text{přiložení}}{\text{náhoda}}\right) = P\left(\frac{A,D}{\text{náhoda}}\right) \times P\left(\frac{B,E}{\text{náhoda}}\right) \times P\left(\frac{C,F}{\text{náhoda}}\right) \quad (3.3)$$

Další výpočet je následující:

$$LR(\text{přiložení}) = \frac{P\left(\frac{A,D}{spol.\text{předek}}\right) \times P\left(\frac{B,E}{spol.\text{předek}}\right) \times P\left(\frac{C,F}{spol.\text{předek}}\right)}{P\left(\frac{A,D}{\text{náhoda}}\right) \times P\left(\frac{B,E}{\text{náhoda}}\right) \times P\left(\frac{C,F}{\text{náhoda}}\right)} \quad (3.4)$$

$$LR(\text{přiložení}) = LR(A,D) \times P(B,E) \times LR(C,F) \quad (3.5)$$

$$\log LR(\text{přiložení}) = \log LR(A,D) + \log LR(B,E) + \log LR(C,F) \quad (3.6)$$

Takto vypočtené log odds jsou dále škálovány a zaokrouhleny k nejbližšímu celému číslu, které potom tvoří pole v matici. Zjednodušeně lze shrnout výpočet skóre v matici následujícím vzorcem [53]:

$$s_{i,j} = \log \frac{q_{i,j}}{p_i p_j}, \quad (3.7)$$

kde $q_{i,j}$ jsou cílové frekvence pro zarovnané páry aminokyselin (jejich výskyt) a $p_i p_j$ pravděpodobnosti jejich výskytu (viz rovnice 3.2 a 3.3).

Zvláštních hodnot pak nabývá diagonála matice. Existuje totiž předpoklad, že na diagonále matice nedochází k mutacím. Skóre na diagonále je pak vytvářeno s ohledem na mnoho aspektů, jako je například: relativní frekvence aminokyseliny ve sloupci, chemické a fyzikální rozdíly mezi aminokyselinami atd.

3.2.2 Substituční matice pro nukleotidové sekvence

Nejjednodušší maticí, a také maticí, z níž pak vycházejí všechny ostatních matice pro ohodnocování podobnosti nukleotidové sekvence, je tzv. matice identity (identity matrix) nebo také IUPAC matice [33]. Tato matice je vhodná pro srovnání dvou blízce příbuzných sekvencí, ale velmi nevhodná pro srovnání dvou vzdálených sekvencí. Jejím hlavním předpokladem je fakt, že jeden nukleotid není schopen mutace v jiný nukleotid. Shody na

diagonále nesou kladnou hodnotu a neshody hodnotu nulovou. Matice identity je znázorněna na obrázku 5.

	A	T	G	C
A	1	0	0	0
T	0	1	0	0
G	0	0	1	0
C	0	0	0	1

Obrázek 5: Matice identity. Převzato a upraveno z [33].

Velmi používaná substituční matice pro hodnocení zarovnání nukleotidů je matice programu BLAST (jinak také nazývá jako matice *nuc44*). BLASTem nejběžněji užívaná matice pro zarovnávání nukleotidů je uvedena na obrázku 6. V této matici je shoda ohodnocena hodnotou 5 a neshoda hodnotou -4. Jedná se o nejmenší možnou matici pro zarovnávání nukleotidů (matice 4x4). Počet řádků a sloupců matice odpovídá počtu bází, které mohou být v nukleotidech za situace, že všechny nukleotidy jsou jasně definované.

	A	T	G	C
A	5	-4	-4	-4
T	-4	5	-4	-4
G	-4	-4	5	-4
C	-4	-4	-4	5

Obrázek 6: Substituční matice programu BLAST – matice *nuc44*. Převzato a upraveno z [4].

Speciálním druhem substituční matice je tranzitní transverzní matice. Tato matice počítá se záměnami jako je tranzice a transverze, jak již vyplývá z jejího názvu. Tranzicí se rozumí bodová mutace, tedy substituce jednoho nukleotidu za jiný, a v případě tranzice jde o záměnu purinové báze za purinovou nebo pyrimidinové za pyrimidinovou [27]. Transverze je také druh bodové mutace, avšak rozdíl oproti tranzici je takový, že jde o záměnu purinu za pyrimidin nebo pyrimidinu za purin. Tranzitní transverzní matice je na obrázku 7.

	A	T	G	C
A	1	-5	-5	-1
T	-5	1	-1	-5
G	-5	-1	1	-5
C	-1	-5	-5	1

Obrázek 7: Tranzitní transverzní matice. Převzato a upraveno z [4].

3.2.3 Substituční matice pro sekvence aminokyselin

Existují dva základní typy substitučních matic pro srovnávání podobnosti sekvencí aminokyselin – matice PAM (Point Accepted Mutation) a BLOSUM (Blocks of Amino Acid Substitution Matrix) [14].

První matice PAM (PAM1) byla vytvořena v 70. letech Margaret Dayhoffovou a spolupracovníky a byla vytvořena na základě porovnání rozdílů mez 71 proteinovými rodinami [39]. Základním předpokladem je fakt, že za určitý definovaný časový úsek dojde k záměně 1% aminokyselin z celé sekvence, přičemž takto zkoumané sekvence musí vykazovat alespoň 85% podobnost. PAM matice byla vytvořena analyzováním mnoha blízce příbuzných sekvencí za použití globálního zarovnání (vznikly tedy extrapolací) a dále sestavena na základě empirického stanovení frekvence jednotlivých substitucí [34]. PAM matic existuje obrovské množství. Po názvu PAM vždy následuje číslo vyjadřující očekávanou míru substituce v případě, že část aminokyselin ze zkoumané sekvence podlehla změnám. Matice s vyšší mírou substitucí vznikají vynásobením PAM mezi sebou (tedy například PAM2 vzniká jako PAM1×PAM1 atd.). Tento poměrně jednoduchý princip výpočtu odráží předpoklad, že k evolučním událostem vedoucím ke změnám v sekvenci dochází stále stejným způsobem a nezávisle. Aktuálně nejvyšší číslo PAM matice je PAM250, která popisuje situaci, kdy na 100 aminokyselin připadá 250 mutací [35]. PAM250 je znázorněna na obrázku 8. Možné však je teoreticky spočítat i další PAM matice. Platí, že čím vyšší číslo PAM matice, tím méně podobné sekvence lze srovnávat. Obecně lze o výhodách PAM matic říci, že je vhodné je používat tam, kde očekáváme malý počet mutací a relativně krátkou evoluční vzdálenost. Nástupci PAM matic jsou například matice Gonnetovy, které jsou vytvořeny pomocí stejného principu jako matice PAM, avšak pro jejich sestavení byl použit výrazně větší soubor výchozích dat [37].

	A	R	N	D	C	Q	E	G	H	I	L	K	M	F	P	S	T	W	Y	V
A	2																			
R	-2	6																		
N	0	0	2																	
D	0	-1	2	4																
C	-2	-4	-4	-5	4															
Q	0	1	1	2	-5	4														
E	0	-1	1	3	-5	2	4													
G	1	-3	0	1	-3	-1	0	5												
H	-1	2	2	1	-3	3	1	-2	6											
I	-1	-2	-2	-2	-2	-2	-2	-3	-2	5										
L	-2	-3	-3	-4	-6	-2	-3	-4	-2	2	6									
K	-1	3	1	0	-5	1	0	-2	0	-2	-3	5								
M	-1	0	-2	-3	-5	-1	-2	-3	-2	2	4	0	6							
F	-4	-4	-4	-6	-4	-5	-5	-5	-2	1	2	-5	0	9						
P	1	0	-1	-1	-3	0	-1	-1	0	-2	-3	-1	-2	-5	6					
S	1	0	1	0	0	-1	0	1	-1	-1	-3	0	-2	-3	1	3				
T	1	-1	0	0	-2	-1	0	0	-1	0	-2	0	-1	-2	0	1	3			
W	-6	2	-4	-7	-8	-5	-7	-7	-3	-5	-2	-3	-4	0	-6	-2	-5	17		
Y	-3	-4	-2	-4	0	-4	-4	-5	0	-1	-1	-4	-2	7	-5	-3	-3	0	10	
V	0	-2	-2	-2	-2	-2	-2	-1	-2	4	2	-2	2	-1	-1	-1	0	-6	2	4

Obrázek 8: Matice PAM250. Převzato a upraveno z [41].

Další velkou skupinou matic pro srovnávání podobnosti aminokyselinových sekvencí jsou matice BLOSUM. Tyto matice byly odvozeny pomocí lokálního zarovnání konzervativních domén (bloků) u relativně vzdálených sekvencí, což je základní rozdíl mezi maticí PAM a BLOSUM [15]. Jsou číslovány stejně jako matice PAM, ovšem číslo zde má zcela jiný význam – vyjadřuje počet konzervativních zbytků ve výchozím souboru sekvencí [36]. Z tohoto faktu tedy vyplývá, že čím vyšší číslo u BLOSUM matice, tím podobnější sekvence (u PAM matic je tomu přesně naopak). BLOSUM matice je výhodné použít u takových sekvencí, kde mohl nastat velký počet mutací, může se jednat o evolučně vzdálené sekvence, avšak předpokládáme zde výskyt evolučně podobných konzervativních sekvencí, které jsou také středem našeho zájmu.

Konstrukce matic BLOSUM je poměrně složitý proces pracující s mnoha parametry z oboru statistiky. Tento proces se dá stručně shrnout v následujících krocích [38]:

1) Průchod databáze s proteinovými rodinami a nalezení výrazně konzervativních bloků. Dále je provedena shluková analýza a takové aminokyseliny, které jsou shodné nad určitý definovaný limit, jsou umístěny do stejného shluku.

2) Zarovnání těchto konzervativních sekvencí bez mezer a výpočet relativních četností jednotlivých aminokyselin a pravděpodobností jejich substituce.
3) Výpočet log-odds skóre pro každou z 210 možných dvojic 20 standardních aminokyselin substituce (počítá se tedy četnost výskytu všech možných dvojic v sekvencích, a to pro všechny aminokyseliny). Je třeba normalizovat jednotlivé četnosti velikostmi shluků, aby nedošlo ke zvýhodnění či naopak znevýhodnění určitých četností.

Nejběžněji používanou BLOSUM maticí je BLOSUM62 a dá se říci, že se jedná o vůbec nejpoužívanější substituční matici vůbec. Je uvedena na obrázku 9.

	A	R	N	D	C	Q	E	G	H	I	L	K	M	F	P	S	T	W	Y	V
A	4																			
R	-1	5																		
N	-2	0	6																	
D	-2	-2	1	6																
C	0	-3	-3	-3	9															
Q	-1	1	0	0	-3	5														
E	-1	0	0	2	-4	2	5													
G	0	-2	0	-1	-3	-2	-2	6												
H	-2	0	1	-1	-3	0	0	-2	8											
I	-1	-3	-3	-3	-1	-3	-3	-4	-3	4										
L	-1	-2	-3	-4	-1	-2	-3	-4	3	2	4									
K	-1	2	0	-1	-3	1	1	-2	-1	-3	-2	5								
M	-1	-1	-2	-3	-1	0	-2	-3	-2	1	2	-1	5							
F	-2	-3	-3	-3	-2	-3	-3	-3	-1	0	0	-3	0	6						
P	-1	-2	-2	-1	-3	-1	-1	-2	-2	-3	-3	-1	-2	-4	7					
S	1	-1	1	0	-1	0	0	0	-1	-2	-2	0	-1	-2	-1	4				
T	0	-1	0	-1	-1	-1	-1	-2	-2	-1	-1	-1	-1	-2	-1	1	5			
W	-3	-3	-4	-4	-2	-2	-3	-2	-2	-3	-2	-3	-1	1	-4	-3	-2	11		
Y	-2	-2	-2	-3	-2	-1	-2	-3	2	-1	-1	-2	-1	3	-3	-2	-2	2	7	
V	0	-3	-3	-3	-1	-2	-2	-3	-3	3	1	-2	1	-1	-2	-2	0	-3	-1	4

Obrázek 9: Substituční matice BLOSUM62. Převzato a upraveno z [40].

3.3 Penalizace mezer

Pomocí substituční matice lze postihnout substituce, které proběhly v průběhu evoluce v sekvenci, avšak nepostihuje ztráty (delece) či přidání (inzerce) nukleotidů či aminokyselin

[13]. Delece a inzerce je však nezbytné také penalizovat, a to pomocí penalizování mezer. Tato penalizace se tedy týká globálního zarovnání, při němž dochází k vnášení mezer. Existují dva typy mezer, které jsou penalizování. První jsou penalizace za otevření mezery (gap open penalty) a druhá je penalizace za rozšíření existující mezery o jeden zbytek (gap extension penalty). Penalizace za otevření mezery je výrazně vyšší, než penalizace za rozšíření mezery. Důvodem je fakt, že pokud již v sekvenci došlo k inzerci či deleci – již tedy existuje mezera – tak její rozšíření na výslednou podobnost nemá příliš velký vliv na rozdíl od vzniku mezery nové.

Pokud by penalizace za mezeru byla příliš velká, algoritmus by takto vložené mezery přeskakoval a nedošlo by tak ke správnému zarovnání. Naopak pokud je zvolená penalizace za mezery příliš malá, dochází k jejich nadbytečnému vkládání – jsou vkládány úplně všude, kde by se dala očekávat neshoda, což ovšem nepřináší takové srovnání, jaké požadujeme.

3.4 Algoritmus Needelman-Wunsch

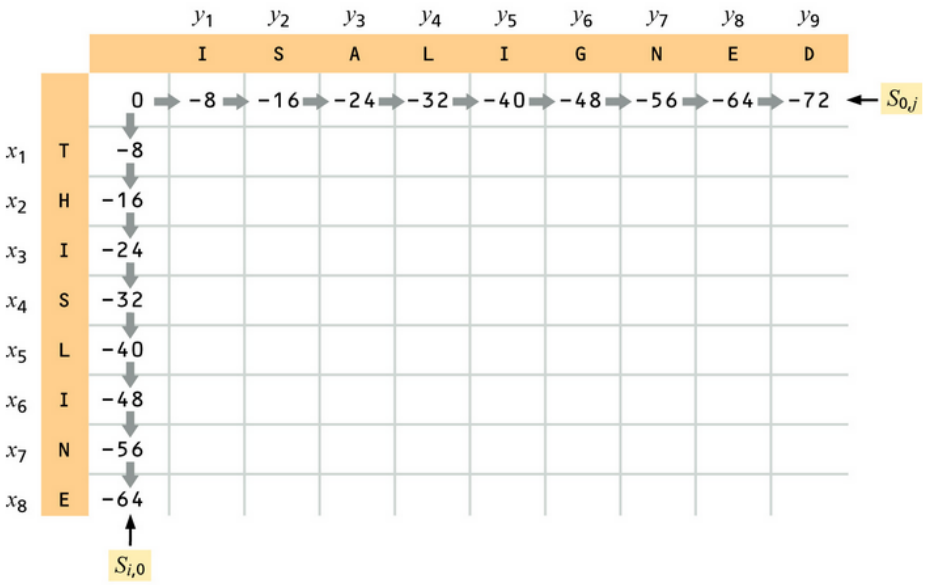
Jedná se o první vyvinutý algoritmus pro tento typ vyhledávání vůbec. Řadí se mezi algoritmy dynamického programování (průběh algoritmu se mění společně s tím, jak se mění problém, který algoritmus řeší). Popsán byl poprvé v roce 1970 a jeho hlavním principem je srovnávání sekvencí nukleotidů/aminokyselin vždy pár po páru (pairwise alignment – srovnání dvou sekvencí) s cílem nalézt maximální podobnost [42]. Každému páru je pak uděleno skóre, které je dáno sumou hodnot podobnosti toho kterého páru (resp. suma hodnot podobnosti jejich nukleotidových/aminokyselinových zbytků) a následným odečtením penalizací za každou vloženou mezeru. Jedná se o algoritmus pro globální zarovnání dvou sekvencí [16]. Princip průběhu algoritmu je možno přiblížit v následujících krocích:

1) Vstupem algoritmu jsou dvě sekvence – dotazovaná a databázová, skórovací matice a nastavení penalizace za vnesení mezer.
2) Prvním krokem je sestavení matice, kde jednotlivé řádky odpovídají znakům jedné sekvence a sloupce znakům sekvence druhé. Celková velikost matice je vždy o jeden řádek a jeden sloupec větší než jsou délky srovnávaných sekvencí. Tento volný řádek a sloupec pak zůstávají volné.
3) V levém horním rohu matice je zapsána hodnota 0 a hodnota každého dalšího pole je dopočítávána dle vzorce:
   $$\max((H(i-1,j)+d),(H(i,j-1)+d),\big(H(i-1,j-1)+M(i,j)\big)) \qquad (3.7)$$
   kde
   $H(i,j)$ – ohodnocení pole v matici na příslušných souřadnicích i, j
   d – hodnota penalizace za vnesení mezery
   $M(i,j)$ – ohodnocení shody či neshody na poli v matici na příslušných souřadnicích i, j

4) Tímto způsobem je postupně nalezen „průchod maximy“ celou maticí a jednotlivé hodnoty jsou zapsány do polí matice.
5) Průchod maticí reverzně od pravého dolního rohu – cílem tohoto kroku je zjistit, jak – tedy z jakých předchozích polí byla získána maximální hodnota.
6) Takto nalezený průchod maticí ukazuje na nejlepší možné globální zarovnání.

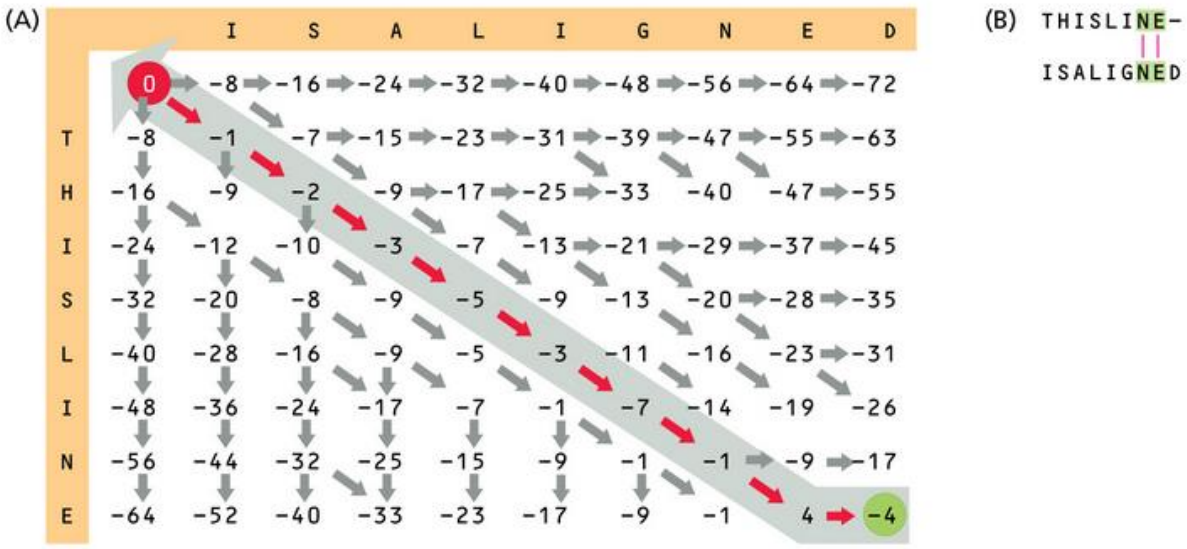
Princip algoritmu Needleman – Wunsch je popsán na konkrétním příkladu [43]:
Úkolem je srovnat dvě sekvence:
`THISLINE`
`ISALIGNED`

Volba parametrů: matice BLOSUM62 a penalizace za vnesení mezer dle vzorce $g(ngap) = -d \cdot ngap$, kde ngap je počet vnesených mezer (v tomto ukázkovém příkladu se nerozlišuje mezera otvírací a mezera rozšiřovací). Hodnota d bude zvolena například -8. Matice pro výpočet algoritmu se začne plnit, jak je znázorněno na obrázku 10.



Obrázek 10: Začátek plnění matice pro algoritmus Needleman – Wunsch. Převzato a upraveno z [43].

Na obrázku 11 je pak znázorněn finální průchod algoritmu Needleman – Wunsch sestavenou maticí:



Obrázek 11: Optimální zarovnání algoritmem Needleman – Wunsch při ohodnocení mezer d = -8. Převzato a upraveno z [43].

3.5 Algoritmus Smith-Waterman

Algoritmus Smith – Waterman pochází z roku 1981 a jedná se o rozšířenou verzi algoritmu Needleman – Wunsch [17]. Díky tomuto faktu jde tedy opět o algoritmus dynamického programování. Rozdíl oproti Needlemanu – Wunschovi je v tom, že je schopen provádět lokální zarovnání, tzn. srovnává mezi sebou pouze blízce příbuzné úseky v sekvencích, ne obě dvě sekvence najednou. Tento algoritmus je tedy schopen detekovat pouze lokální podobnost i přesto, že zbytek sekvence bude zcela odlišný.

Algoritmus Smith – Waterman „ctí“ zásady jako je skórovací matice či penalizace mezer, stejně jako Needleman – Wunsch, avšak s tím rozdílem, že buňky ve skórovací matici, které by byly vypočteny, že nesou zápornou hodnotu, jsou nastaveny na nulu. Právě toto opatření způsobuje dobrou viditelnost lokálních zarovnání, která jsou ohodnocena kladně.

Začátek algoritmu je zcela stejný jako u algoritmu Needleman – Wunsch (nastavení velikosti matice, začátek od levého horního rohu, kde je hodnota nula, dopočítávání následujících polí). Rozdíl nastává ve fázi zpětného průchodu maticí – začíná v buňce matice, která je ohodnocena nejvyšším skóre a postupuje, až narazí na buňku ohodnocenou nulou – tato cesta pak ukazuje nejlepší lokální zarovnání a může být tedy kratší než globální zarovnání (právě díky „vynulování“ záporně ohodnocených polí v matici).

4 Programy a editory pro práci s homologními geny

Tyto programy jsou založeny na konzervativním charakteru sekvencí, které mají určitou definovanou funkci. Využívají metod jak lokálního, tak globálního zarovnávání. Jsou schopny identifikovat pouze takové geny, které se již nacházejí v databázi. Algoritmy využívající lokální zarovnání typicky využívá asi nejznámější a nejužívanější komerční program – BLAST. S globálním zarovnáním pracují programy jako je např. LAGAN.

4.1 BLAST

BLAST (Basic Local Alignment Search Tool) je program, který srovnává primární biologické sekvence [22]. Je schopen srovnat jak sekvence nukleotidové, tak sekvence aminokyselin, případně různé kombinace sekvenci (viz. tabulka 1). Jedná se o nejznámější program, který umožňuje uživateli srovnat dotazovanou sekvenci se sekvencí uloženou v databázi a stanovit její případnou homologii.

Byl navržen v roce 1990 pod záštitou NCBI a prvotní návrh pracoval na bázi algoritmu Smith-Waterman. Smith-Waterman algoritmus zajistil programu značnou přesnost, avšak na úkor rychlosti a s požadavkem vysokého výpočetního výkonu. V současné době se jedná o program poměrně citlivý, avšak nelze u něj zaručit zcela přesné srovnání dotazu se sekvencí s databáze. Výhodou naopak je značná rychlost a menší nároky na výpočetní výkon, což umožňuje prohledávání dokonce i rozsáhlý genomů. Značným kladem BLASTu je také to, že se jedná o aplikaci volně přístupnou na stránkách NCBI. Samozřejmě je také možné si BLAST stáhnout zdarma do svého počítače jako „blastall“ a pracovat pak pomocí příkazového řádku. BLAST je šířen jako open-source program, což umožňuje neustálé vylepšování programu.

4.1.1 BLAST algoritmus

Na vstupu celého algoritmu jsou vstupní data, a to buďto ve formátu FASTA nebo ve formátu GenBank, a substituční matice [20]. Výstupní data při používání BLASTu přímo ze stránek NCBI jsou typicky ve formátu HTML a běžné také je grafické znázornění výsledků, a to ve formě tabulek a grafů znázorňujících shody. Pro uživatele je grafické znázornění jednoznačně nejlepší pro orientaci ve výsledcích. Možný je však také výstup ve formátu XML či ve formě prostého textu.

Hlavní myšlenkou celého algoritmu je nacházení takzvaných high-scoring segment pairs (HSP; segmentové páry s vysokým skóre, avšak český překlad se prakticky nepoužívá).

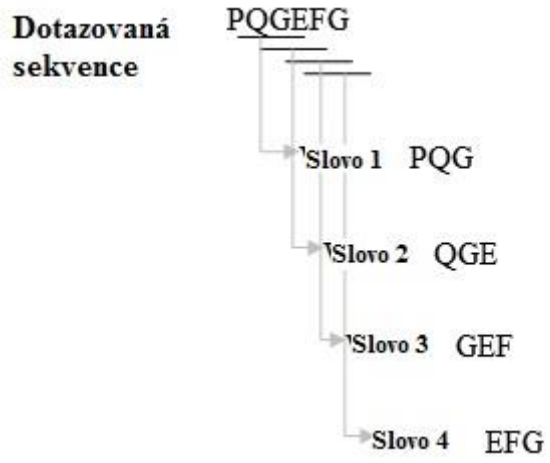
HSP jsou takové páry v sekvenci, které již mají pro srovnávání statistický význam. BLAST vyhledává sekvence s vysokým skóre po srovnání dotazované sekvence a sekvence z databáze. Využívá k tomu heuristické metody částečně podobné algoritmu Smith-Waterman, avšak právě díky heuristickému přístupu je BLAST algoritmus až 50x rychlejší než „klasický“ Smith-Waterman algoritmus [23].

BLAST tedy nesrovnává obě dvě sekvence po celé jejich délce, ale pouze hledá polohu krátkých shod mezi sekvencemi. Proces, při němž jsou hledána počáteční slova (tato slova se skládají z jasně definovaných sad znaků – písmen; např. pro nukleotidovou sekvenci DNA jsou to písmena značící jednotlivé báze – A, T, C a G) se nazývá „seeding“ (česky by se dalo vyjádřit jako „zasetí“ algoritmu). Ve chvíli, kdy algoritmus nalezne první shodu, začíná postupné lokální zarovnávání. BLAST nejprve z dotazované sekvence odstraní úseky s malou mírou příbuznosti a úseky repetitivní [19]. Poté je vytvořen seznam úseků dotazované sekvence, který označuje délku sekvence (k). Pro DNA bývá typicky $k = 11$ a pro proteiny $k = 3$. Sestavený seznam pak obsahuje veškeré varianty úseků dotazované sekvence. V sekvenční databázi jsou poté jednotlivé úseky hledány a správnost nalezení je hodnocena pomocí substituční matice. Pomocí ní jsou nalezeny HSP. Zarovnání je následně rozšířeno na obě strany směrem od HSP za využití stále stejné substituční matice.

Vylepšené verze BLASTu pak počítají i s případy vnesení mezer. V těchto případech je pak již třeba využít metod dynamického programování.

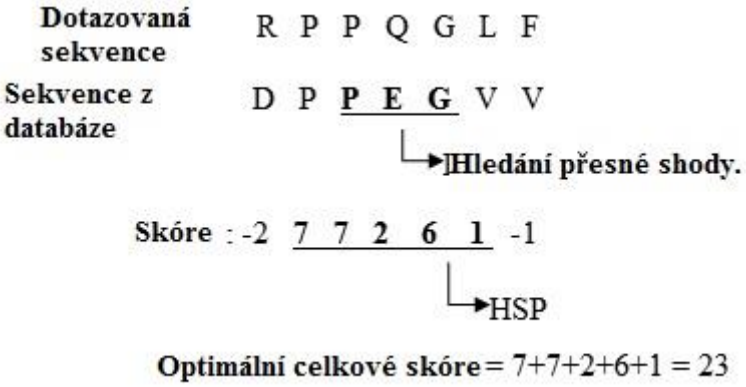
Příklad BLAST na algoritmu při srovnání protein-protein (tedy za použití blastp) [23]:

1) Odstranění úseků s malou mírou příbuznosti a úseků opakujících se – úseky s malou mírou příbuznosti jsou takové, které jsou složeny z několika druhů prvků. Problémem u nich je to, že po průchodu substituční maticí mohou poskytovat velmi vysoké skóre, což ovšem znemožňuje programu najít skutečně významné shody. Takovéto sekvence jsou v případě nukleotidové sekvence označeny písmenem N a v případě sekvence aminokyselin označeny písmenem X. BLAST takto značené sekvence pak ignoruje. Existují také různé programy na odfiltrování oblastí s nízkou mírou příbuznosti (např. SEG pro proteinové sekvence a DUST pro nukleotidové).
2) Vytvoření k-písmenného seznamu slov z dotazované sekvence – principem je tvoření postupného seznamu písmen o dané délce k tak dlouho, než je „přečtena“ celá sekvence a nalezena veškerá slova o dané délce k. Graficky je tento proces znázorněn na obrázku 12.



Obrázek 12: Metoda sestavení *k*-písmenného seznamu slov z dotazované sekvence. Zde je hledaná délka slova $k = 3$, což je délka charakteristická pro proteiny. Převzato a upraveno z [23].

3) Vytvoření seznamu možných shod ve slovech – v tomto kroku je porovnáváno vždy jedno slovo se všemi možnými slovy vytvořenými v kroku 2). Podle přesnosti shody je pak vždy každému srovnání uděleno určité skóre. Skóre je vytvořeno pomocí průchodu přes substituční matici. Zároveň je také definováno prahové skóre *T*, pomocí nějž je redukován počet odpovídajících shod ve slovech. Prakticky tedy takové dvojice slov, které dosáhnou skóre stejné či vyšší než je *T*, zůstávají v seznamu možných shod, avšak ty s nižším skóre než je *T* jsou z tohoto seznamu vyřazeny
4) Uspořádání slov, která zbyla v seznamu možných shod, do vyhledávacího stromu – sestavení takového stromu umožňuje rychlé srovnání slov z dotazu se slovy z databáze.
5) Postupně je třeba opakovat 3) a 4) pro každé *k*-písmenné slovo z dotazované sekvence.
6) Prohledávání sekvence z databáze pro nalezení shody se zbývajícími slovy ze seznamu – jakmile je nalezena přesná shoda slova mezi sekvencí z databáze a dotazovanou sekvencí, je tato shoda použita k seedingu možného zarovnání bez mezer mezi oběma sekvencemi.
7) Rozšíření přesných shod na HSP – od místa nalezení přesné shody dochází k rozšiřování zarovnání ve směru vlevo i vpravo tak dlouho, dokud skóre HSP nezačne klesat. Názorný příklad tohoto rozšiřování je uveden na obrázku 13.



Obrázek 13: Proces rozšiřování přesných shod. Převzato a upraveno z [23].

8) Vyhodnocení seznamu všech HSP z databáze, jejichž skóre je dostatečně vysoké – v tomto kroku dojde k vypsání všech HSP, jejich hodnota je vyšší než hodnota skóre *S*, což je cut-off hodnota (jde tedy o hodnotu, od níž považujeme sekvence za příbuzné či homologní).
9) Vyhodnocení HSP skóre – v tomto kroku je potřeba statisticky vyhodnotit význam HSP skóre. K tomuto vyhodnocení je využito tzv. Gümbelovo extrémní rozdělení hodnot (EVD). V souladu s EVD pak pravděpodobnost *p* pozorování skóre *S* je rovna *x* – vztah je dán následující rovnicí:

$$p(S \geq x) = 1 - \exp(-e^{-\lambda(x-\mu)}) \tag{4.1}$$

kde

$$\mu = \frac{\log(Km'n')}{\lambda} \tag{4.2}$$

Statistické parametry λ a K jsou odhadovány (do podoby Gümbelova EVD) pomocí vhodného rozdělení skóre lokálního zarovnání bez mezer, dále pomocí sekvence dotazu a mnoha různých verzí sekvence v databázi. λ a K závisí na volbě substituční matice, na hodnotě penalizace za vnesení mezer a na složení sekvence (tedy na tom, s jakou frekvencí se jednotlivé znaky objevují). Výrazy *m'* a *n'* vyjadřují efektivní délky sekvencí – dotazované a databázové. Originální délky sekvencí jsou totiž zkráceny do podoby těchto efektivních délek, a to z důvodu tzv. kompenzace účinku hrany – pokud by došlo k zarovnávání na začátku jedné ze sekvencí, je velmi pravděpodobné, že by sekvence nebyla dostatečně dlouhá pro optimální zarovnání. Tyto efektivní délky *m'* a *n'* mohou být vypočteny následovně:

$$m' \approx m - \frac{\ln Kmn}{H} \tag{4.3}$$

$$n' \approx n - \frac{\ln Kmn}{H} \tag{4.4}$$

kde H je průměrné očekávané skóre pro zarovnaný zbytkový pár po zarovnání dvou náhodných sekvencí. Dle [20] byly jednotlivé parametry při lokálním zarovnání bez mezer a při použití matice BLOSUM62 stanoveny následovně: $\lambda = 0{,}318$; $K = 0{,}13$ a $H = 0{,}40$.

Očekávané skóre E shod s databází představuje počet opakování, při nichž by nesouvisející databázová sekvence získala náhodně vyšší skóre S než je x. Očekávanou hodnotu E získanou po prohledání databáze, která obsahuje D sekvencí, vyjadřuje tato rovnice:

$$E \approx 1 - e^{-p(s>x)D} \tag{4.5}$$

V případě, kdy $p<0,1$, může se E blížit Poissonovu rozložení:

$$E \approx pD \tag{4.6}$$

Hodnota E (jinak také E-value), která je při výstupu z BLASTU jedním z klíčových údajů, tedy vyhodnocuje význam HSP skóre při lokálním zarovnání bez mezer.

10) Vytvoření dvou či více HSP oblastí pro delší zarovnání – při delším zarovnání lze najít v sekvenci dva či více HSP regionů. Tato informace je dalším potvrzením podobností mezi sekvencí dotazovanou a sekvencí z databáze. Pro správné vyhodnocení této situace existují dva možné přístupy – Poissonovo rozdělení a tzv. metoda „suma skóre“. Předpokládejme, že existují dvě spojené HSP oblasti s páry o následujícím skóre: pár 1 – 70, 30 a pár 2 – 50, 40. Poissonovo rozdělení považuje za významnější pár 2, jelikož má maximální nižší skóre – 40>30. Oproti tomu metoda „suma skóre přisuzuje větší význam páru 1, a to z důvodu většího celkového součtu skóre – 70+30>50+40. Původní BLAST využívá Poissonovo rozdělení, BLAST počítající s mezerami používá metodu „suma skóre”.
11) Zobrazení lokálního zarovnání s mezerami pomocí algoritmu Smith-Waterman – původní BLAST počítal pouze se zarovnáním bez mezer. BLAST2 vytváří jediné zarovnání s mezerami, které může obsahovat veškeré původně nalezené HSP regiony.
12) Vypsání všech shod, které dosáhly menší hodnoty než je prahová hodnota čísla E (pokud $E < 0{,}02$, dá se předpokládat, že nalezené sekvence jsou skutečně homologní; pokud $E > 1$, nalezená shoda je pouhým dílem náhody).

Kromě E-value se na výstupu programů srovnávajících dvě sekvence setkáváme ještě s tzv. Z-score, což je hodnota vyjadřující jak moc nepravděpodobná je nalezená shoda – tedy čím vyšší bude Z-score, tím jistěji můžeme usoudit, že vytvoření srovnání není pouze dílem náhody [43]. Při Z-score > 3 se již dá usuzovat, že se bude jednat skutečně o homologii. Naopak další statistická hodnota P-value popisuje pravý opak, čím vyšší je, tím větší pravděpodobnost, že nalezená shoda je dílem náhody. P-value může nabývat hodnot v intervalu [0, 1].

Tabulka 3: Typy BLASTu a jejich význam. Převzato a upraveno z [11].

Typ programu BLAST	**Popis programu**
Blastn	Dotazovaná sekvence je nukleotidová, srovnávána je taktéž s nukleotidovou sekvencí z databáze.
Blastp	Dotazovaná sekvence je proteinová a srovnávána je s proteinovou sekvencí z databáze.
PSI-BLAST (Position Specific Iterative BLAST), blastpgp	Umožňuje vyhledávání velmi vzdálených proteinových sekvencí. Automaticky generuje pozičně specifickou substituční matici.
Blastx	Dotazovaná sekvence je nukleotidová a srovnávána je s proteinovou sekvencí z databáze.
Tblastx	Dotazovaná sekvence je translatovaná nukleotidová a srovnávána je s přeloženou nukleotidovou sekvencí z databáze. Jedná se o nejpomalejší typ BLASTu. Je možné jím srovnávat velice vzdáleně příbuzné sekvence.
Tblastn	Dotazovaná sekvence je proteionová a srovnávána je s nukleotidovou sekvence z databáze, a to se všemi možnými čtecími rámci.
Megablast	Slouží k porovnávání velkého počtu dotazovaných sekvencí. Funguje jako samostatný příkazový řádek „megablast“. Principem je to, že megablast dokáže zřetězit více vstupních sekvencí ještě předtím, než dojde k prohledávání databáze. Následně jsou analyzovány výsledky vyhledávání pro zisk jednotlivých zarovnání a také jsou hodnoceny ze statistického hlediska.

4.2 PatternHunter

PatternHunter je komerční software pro vyhledávání homologních sekvencí a vytvořen byl v roce 2002 [18]. Jeho největší výhodou je, například oproti BLASTu, větší rychlost výpočtu. Právě díky tomuto zlepšení je PatternHunter využitelný v oblasti srovnávací genomiky, kdy je cílem srovnat opravdu velké úseky genetické informace, jako jsou například celé chromozomy. Pro výpočet takto velkých úseků BLAST a jiné podobné programy potřebují velmi mnoho času a také velkou paměťovou kapacitu. PatternHunter například zvládl srovnat genom myši s lidským genom za pouhých 20 CPU dnů (plné využité procesoru po dobu dvaceti dnů), zatímco BLAST by na tuto akci potřeboval minimálně 20 CPU let, a to se stejnou citlivostí.

Důvodem zvýšené rychlosti je lepší metodika optimálně rozmístěných semínek (seeds – jedná se o určité úseky sekvence, v podstatě jde o malé vyhledávací řetězce). Princip je založen na tom, že je zbytečně složité srovnávat každou pozici dotazu s každou pozicí databáze vzhledem k tomu, jak rozsáhlý může být dotaz a jak rozsáhlé jsou databáze. Citlivost, s jakou program vyhledává shody, je také do určité míry ovlivně vzdáleností, s jakou jsou od sebe jednotlivé sousední vyhledávací řetězce. Velké řetězce (seeds) nejsou schopny najít izolované homologie, zatímco ty příliš malé naházejí i ty nejnepatrnější shody, což ovšem zase výrazně prodlužuje výpočet. Dále je výhodou to, že seeds u PatternHunteru mohou být složena z přerušených sekvencí. Základem úspěchu nalezení optimálního poměru mezi dobou výpočtu a citlivosti je tedy správné nastavení seeds.

Stejně jako u BLASTu se zde setkáváme s *k*-písmenným seznamem slov, avšak oproti BLASTu je dosti alternativní (BLAST používal tento *k*-písmenný seznam postupně jako jednotlivá seeds). První fází programu je jakési filtrování, kdy jsou vyhledávány (doslova loveny – anglicky „hunt“) takové *k* alternativní shody, které naznačují, že by se mohlo jednat o ten nejvýhodnější vzor (anglicky „pattern“ – právě odsud pochází název celého programu PatternHuntern, doslovně tedy „lovec vzorů“). Další fází je samotné zarovnávání, které je totožné s BLASTem. Jediným rozdílem je možnost použití více seeds najednou, což zvyšuje citlivost bez toho, aniž by byla snížena rychlost.

Pomocí PatternHunteru lze vyhledávat homologie DNA – DNA i protein – protein. Ve srovnání s BLASTem a megablastem bylo zjištěno, že PatternHunter je přibližně 100x rychlejší a základ, z něhož vychází, tedy algoritmus Smith – Waterman, rychlostní přesáhl již 3000x [44].

4.3 AVID

AVID je typ programu, který umožňuje srovnávat pouze homologní sekvence, a to ještě s takovým omezením, že v sekvenci se nesmí nacházet žádné duplikace, inverze či translokace [25]. Je vhodný pro srovnávání dvou poměrně blízkých genomů, jako je například člověk – šimpanz či člověk – myš (jde vždy o zástupce savců) nebo pro srovnání dvou různých druhů bakterií z jednoho rodu – např. *Streptococcus pyogenes* a *Streptococcus agalactiae*. AVID nejprve detekuje lokální podobnosti mezi dvěma sekvencemi, poté je vybere a přesně stanoví jejich polohu a seskupí je do uspořádané množiny místních podobností (tomuto kroku se říká „kotva“ – „anchor“ a tedy celý krok je označován jako „anchoring“ – „ukotvení“). V poslední fázi srovná proložené oblasti sekvence. Velice významná je právě fáze „anchoringu“, a to hlavně z toho důvodu, že snižuje výpočetní čas tím, že nenutí program pracovat s velkou, a tedy problematickou sekvencí, ale rozdělí ji do mnoha sekvencí menších. Pro stanovení správných „kotev“ je nezbytné nejprve připravit a identifikovat odpovídající si regiony.“Anchoring“ je pochopitelně tím snazší, čím podobnější si dvě sekvence jsou.

4.4 LAGAN

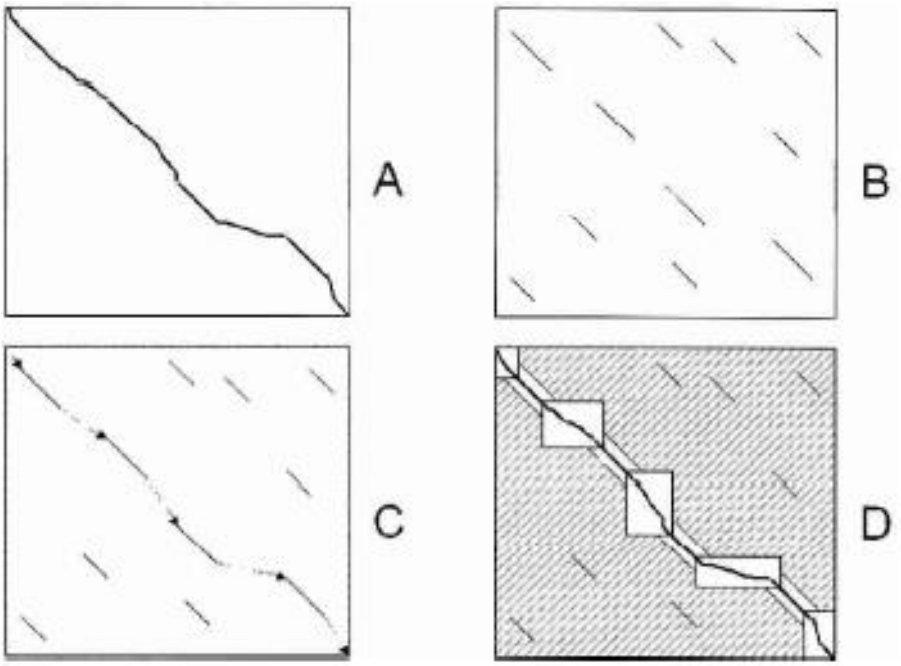
Jedná se o program využívající globální zarovnání a je veřejně dostupný na této internetové adrese: http://lagan.stanford.edu.

LAGAN je program, který umožňuje srovnávat dvě poměrně dlouhé a velice málo příbuzné sekvence [24]. Je to program přesný, avšak přesnost je úměrná času srovnání. LAGAN většinou pracuje v kombinaci s programem využívajícím lokální zarovnání dlouhých sekvencí (například s BLASTem). LAGAN pracuje na bázi tzv. CHAOS algoritmu, což je metoda pro velice citlivé lokální zarovnání na základě mnoha krátkých, ne zcela přesných slov (na rozdíl od jednoho dlouhého, přesného slova) [26]. LAGAN nejprve použije CHAOS rekurzivně v místech sekvence, kde je malá koncentrace kotev tak, aby každé dvě po sobě následující dvojice kotev byly od sebe vzdáleny méně než je maximum vzdálenosti, a poté je použit algoritmus dynamického programování v oblastech kolem kotvy.

Celý postup, jak program pracuje, lze stručně shrnout do tří základních kroků:

1) vytvoření lokálních zarovnání mezi dvěma sekvencemi
2) vytvoření hrubé globální mapy pomocí seskládání podmnožin lokálních zarovnání
3) výpočet konečného globálního zarovnání pomocí nalezení toho nejlepšího lokálního zarovnání, které zůstane v omezeném prostoru hrubé globální mapy.

Pro názornější ukázku je postup zarovnávání LAGANem znázorněn na obrázku 14:



Obrázek 14: Algoritmus použitý v programu LAGAN. (A) Globální zarovnání mezi dvěma sekvencemi je znázorněno jako cesta mezi levým horním a pravým dolním rohem jejich srovnávací matice. (B) LAGAN nejprve najde všechny lokální podobnosti mezi dvěma sekvencemi a každé z nich přiřadí určitou váhu. (C) LAGAN vypočítá maximální skóre uspořádané podmnožiny lokálních podobností a sestaví dohromady hrubou globální mapu. Dvě místní podobnosti mohou být zřetězeny pouze tehdy, pokud konec jedné sekvence předchází začátek sekvence druhé, a to v obou směrech. (D) LAGAN omezí vyhledávání optimálního zarovnání na oblast kolem „kotev" a vypočítá optimální Needleman-Wunsch zarovnání právě v této oblasti. Převzato a upraveno z [24].

Možnou modifikací LAGANu je M-LAGAN. M-LAGAN v první fázi přiřadí více příbuzných genomů a následně přiřazuje genomy fylogeneticky vzdálené (pracuje tedy na principu mnohonásobného globálního přiřazení). M-LAGAN pracuje ve dvou základních krocích:

1) postupná fáze zarovnávání, během níž dojde k vytvoření mnohonásobného zarovnání, a to buďto pomocí postupného zarovnávání dvou sekvencí nebo skládáním mezilehlých zarovnání vzniklých pomocí algoritmu, který používá klasický LAGAN
2) fáze, při níž lze provést volitelné iterativní zlepšení, což způsobí postupné odstranění každé sekvence z mnohonásobného zarovnání a postupně je přeskupuje společně se zbytkem zarovnání. Toto se děje tak dlouho, dokud není pozorováno žádné významné zlepšení výsledků.

Pomocí mnohonásobného zarovnání lze velice dobře sledovat „kroky evoluce“, a je tedy velmi vhodným programem pro možnou konstrukci fylogenetických stromů z celých genomů. Kvůli zmírnění časové a výpočetní náročnosti tento program využívá heuristických metod, speciálně pak tzv. progresivního zarovnávání (někdy se lze také setkat s pojmem „postupné zarovnávání“). Toto progresivní zarovnání je vytvářeno jako sled po sobě následujících párových zarovnání. Stejný typ zarovnání využívá také program ClustalW.

Dalším “typem” LAGANu je S-LAGAN, který má oproti předchozím dvěma schopnost detekovat různá genomová přeskupení a inverze. Používá se pro globální přiřazení kompletních sekvencí genomu a poskytuje přiřazení všech kombinací vložených sekvencí.

4.5 Clustal

Clustal patří mezi zástupce programů sloužící k mnohočetnému (vícenásobnému) zarovnání, což je takové přiřazení, které slouží k identifikaci konzervovaných sekvenčních motivů a prakticky se realizuje srovnáním tří a více sekvencí v jediném okamžiku. Jedná se o program použitelný pro globální i lokální zarovnání [13].

Clustal patří mezi vůbec nejstarší program svého druhu. Už v roce 1980 byl distribuován poštou na disketách [6]. Původně byl napsán v dnes již nepoužívaném jazyce Microsoft FORTRAN pro MS – DOS. Běžel na počítačích IBM jako čtyři samostatné programy Clustal 1 – Clustal 4. Později byly tyto čtyři programy sloučeny do jediného (Clustal V) a přepsány do jazyka C.

Současné verze Clustalu – Clustal X (potažmo jeho verze Clustal W s rozhraním pro Windows) patří mezi jeden z nejpoužívanějších programů pro mnohočetné přiřazení a je napsán v C++ [5]. Program je možné spustit online ze stránek EBI: http://www.ebi.ac.uk/tools/clustalw2. Zdrojový kód pro PC se systémem Windows, Linux a Macintosh je pak k dispozici na FTP serveru EBI: ftp://ftp.ebi.ac.uk/pub/software/clustalw2/.

Aktuální verze Clustalu je verze ClustalW 2.0. Tato verze programu byla přepsána v C++ a její velkou výhodou je možnost jednoduchého modelu objektu, díky čemuž lze snáze modifikovat některý ze zarovnávacích algoritmů. Díky zařazení nového kódu UPGMA (Unweighted Pair Group Method using Arithmetic – shluková analýza – distanční metoda, jak sestavovat stromy podobností, dřívější verze používají metodu Neighbour Joining guide tree) lze provádět zarovnávání extrémně velkých dat mnohem rychleji. Na standartních stolním PC lze provést zarovnávní až 10 000 sekvencí za minutu s využitím kódu UPGMA. Mnohem uživatelsky příznivější je také grafické zpracování.

Stručně shrnutý algoritmus, který Clustal používá, lze shrnout takto [45]:

1) Clustal nejprve provede zarovnání po párech sekvencí (klasické pairwise alignment) a u každého páru uvede jejich podobnost (odvozenou z matice normalizovaných hodnot vzdáleností jednotlivých párů).

2) Na základě těchto podobností je pak provedena shluková analýza a sestaven dendrogram (tzv. Neighbour Joining guide tree), v němž jsou uspořádány jednotlivé sekvence od nejpodobnější po nejméně podobnou.
3) Poslední krokem je globální přiřazení sekvencí ležících v dendrogramu hned vedle sebe, přičemž další sekvence jsou přiřazovány v souladu s pořadím v dendrogramu.

Jistým úskalím použití programu Clustal je fakt, že umí mezery pouze zavádět, ale již je nedokáže odstranit. Pokud tedy při analýze výsledků narazíme na situaci, kdy se vedle sebe vyskytuje větší počet mezer blízko u sebe, lze předpokládat, že se jedná o chybu. Možností, jak výskyt této chyby omezit, je správně nastavit vstupní parametry. Pokud ani toto nepomůže, je nezbytné parametry nastavit ručně [46].

5 Realizace programu

Praktickým výstupem této diplomové práce je program s uživatelským rozhraním, které umožňuje uživateli vkládat vstupní data (sekvence nukleotidů či aminokyselin), a to buďto ručním vepsáním sekvence (raw formát) nebo vložením kódu sekvence nahráním z databáze. Při nahrávání sekvencí z databáze jsou data ve vstupním formátu FASTA – přípona .ffn pro nukleotidy a .faa pro proteiny. Díky vložení sekvencí ve FASTA formátu jsou odděleny sekvence jednotlivých genů. Nebylo tedy třeba tento problém v programu ošetřovat.

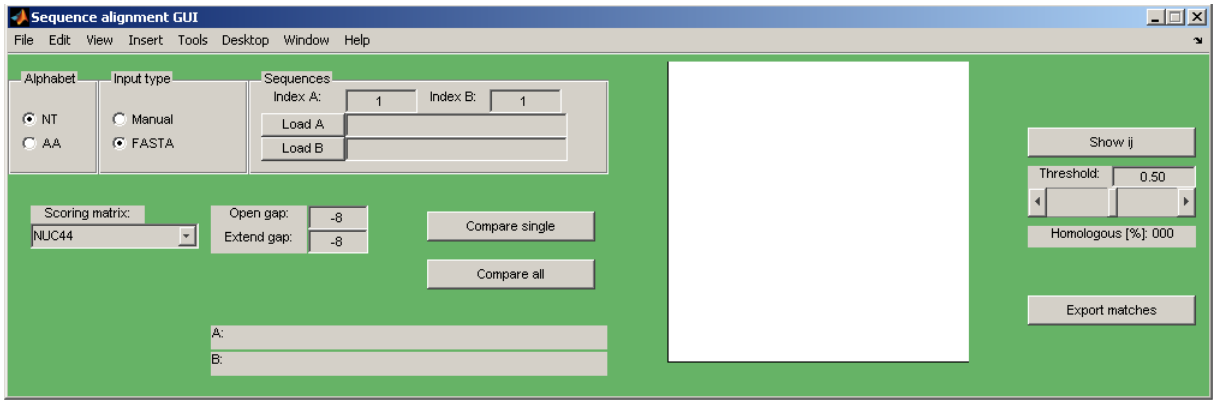
Jádro celého algoritmu je založeno na globálním zarovnání, tedy na algoritmu Needleman-Wunsch. Důvodem volby globálního zarovnání na rozdíl od lokálního bylo výrazně jednodušší normalizování skóre vzniklého zarovnáním.

Výstupem programu po volbě „Compare all“ je bodový diagram, na němž jsou znázorněny shody v sekvencích nalezené po nastavení příslušného thresholdu. Bodový diagram se může kontinuálně měnit společně se změnou nastavení thresholdu. Dalším možným výstupem je dialogové okno, v němž jsou vypsány jednotlivé úseky kódující geny ze sekvence A a k nim nalezené homologie ze sekvence B. Poslední možností je zobrazení „průběhu“ zarovnání – v novém okně jsou pod sebou znázorněny obě sekvence a způsob, jakým jsou zarovnané společně s počtem a procentuálním vyjádřením nalezených shod.

Program je aktuálně omezen pouze na vkládání sekvencí prokaryot, a to z toho důvodu, že prokaryotický genom neobsahuje nekódující oblasti introny, ale pouze kódující exony. Introny by tedy bylo nezbytné, v případě vložení eukaryotického genomu, ještě vystřihnout ze sekvence.

Program byl realizován v prostředí programu Matlab 7.7.0 (R2008b).

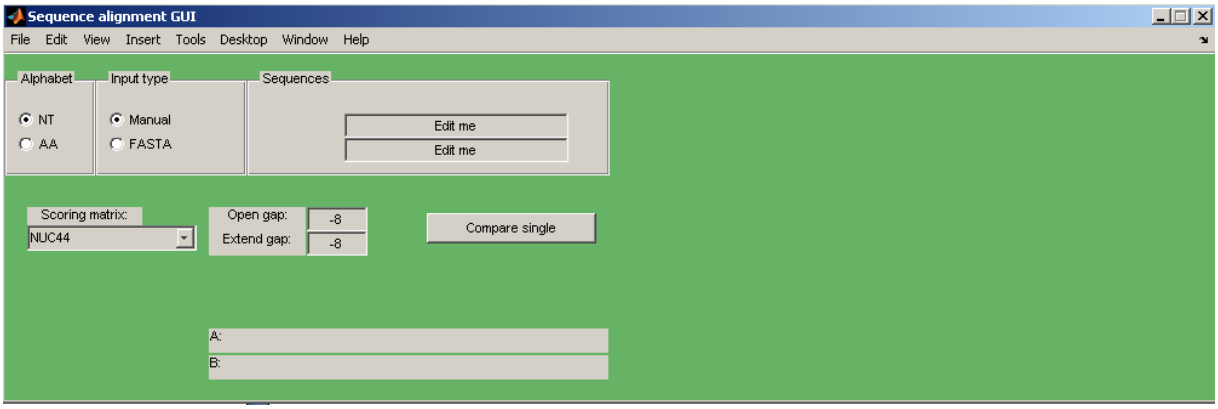
5.1 Uživatelský průvodce



Obrázek 15: Uživatelské rozhraní programu pro srovnávání sekvencí.

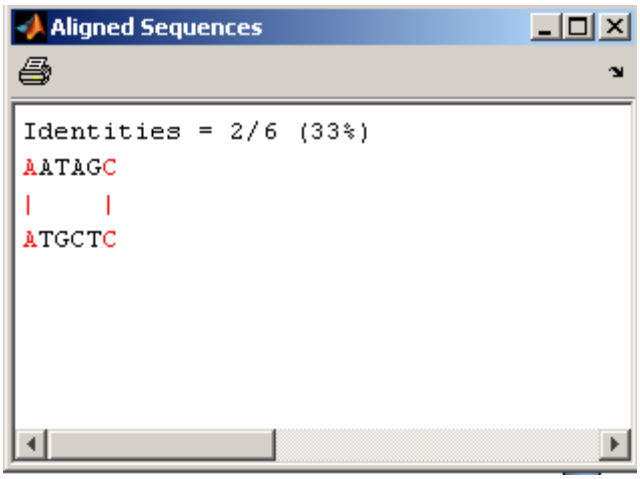
Prvním krokem po spuštění programu je volba typu sekvence, které chce uživatel srovnávat. Na výběr je mezi nukleotidy (v programu jako „NT“) a aminokyselinami („AA“). Tato volba rozhoduje o tom, jaká bude zvolena programem abeceda pro zarovnávání. Jiné znaky jsou v abecedě použity při zarovnávání sekvencí nukleotidů a jiné aminokyselin (o jaké je znaky jde, je uvedeno v tabulce 1 a 2).

Po zvolení typu sekvence uživatel zvolí, zdali chce sekvenci zadat ručně či nahrát z databáze. Pokud uživatel zvolí zadání sekvence ručně, vypadá uživatelské rozhraní následovně:



Obrázek 16: Uživatelské rozhraní programu pro srovnávání sekvencí při volbě manuálního vložení sekvence.

Vzhledem k tomu, že se jedná o jednoduché srovnání dvou sekvencí, není třeba vykreslovat bodový diagram a výsledek se uživateli zobrazí v novém okně tak, jako je znázorněno na obrázku 17. Výstupem je vypsaná procentuální míra podobnosti a způsob, jakým bylo přiřazení provedeno, včetně vnesených mezer.



Obrázek 17: Příklad výstupu při zarovnání manuálně zadaných sekvencí.

Pro volbu nahrání sekvencí z databáze je v programu tlačítko „Load A/B“ – uživatel ze souboru vybere příslušnou sekvenci. Defaultně je jako typ nahrávaného souboru nastaveno „All Fasta File“, přičemž pro nukleotidy je přípustný formát .ffn a pro aminokyseliny .faa.

pokud uživatel zvolí sekvence v jiném formátu než je .ffn a .faa, v Command Window bude hlášen „error“ a zarovnání neproběhne.

V případě, že uživatel zvolí správný formát dat, sekvence se nahrají a jejich název je vypsán do textového boxu.

Další možnou volbou je volba substituční matice. Při zvolení zarovnávání nukleotidové sekvence jsou na výběr pouze matice pro nukleotidové sekvence, tedy konkrétně (bližší informace o maticích vhodných pro srovnání nukleotidů jsou uvedeny v kapitole 3.2.2):

1) Nuc44 – defaultní matice pro zarovnávání, matice je zobrazena na obrázku 6.
2) Identity – matice identity zobrazena na obrázku 5.
3) TTM – tranzitně transverzní matice – uvedena na obrázku 7.
4) MY_SCORES – uživatel si může hodnoty pro shodu a neshodu nastavit sám v novém objevivším se textovém poli.

Pokud uživatel zvolí zarovnávání aminokyselinových sekvencí, je na výběr z následujících typů matic (více o nich v kapitole 3.2.3):

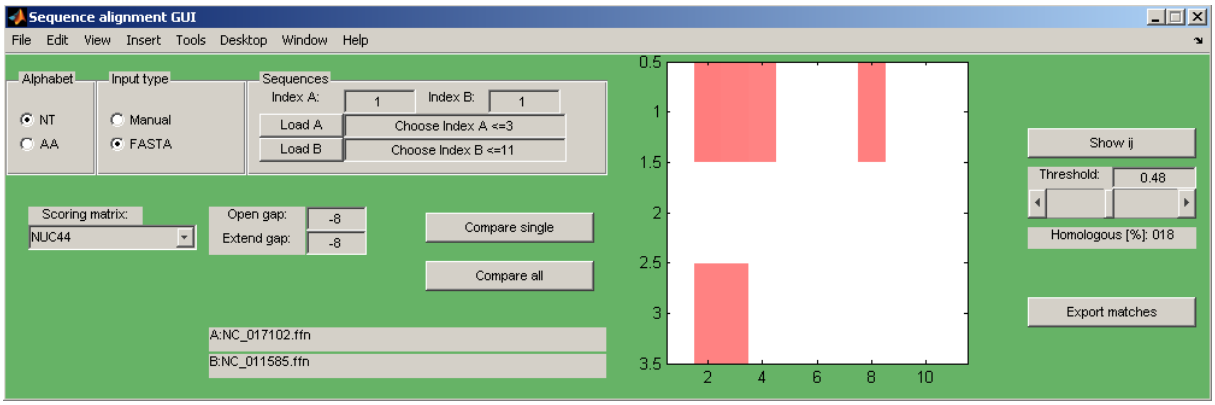
1) BLOSUM – na výběr jsou různé typy této matice dle počtu konzervativních zbytků – konkrétně tedy BLOSUM35, 40, 45, 50, 55, 60, 62, 65, 70, 75, 80, 85, 90, 100.
2) PAM – na výběr jsou různé typy této matice dle počtu očekávaných substitucí – konkrétně tedy 10 – 500 s krokem 10.

Dále si uživatel může zvolit hodnotu, s níž bude penalizováno vnesení mezery – otevírací i rozšířené. Defaultní hodnota je nastavena na -8, což je standard.

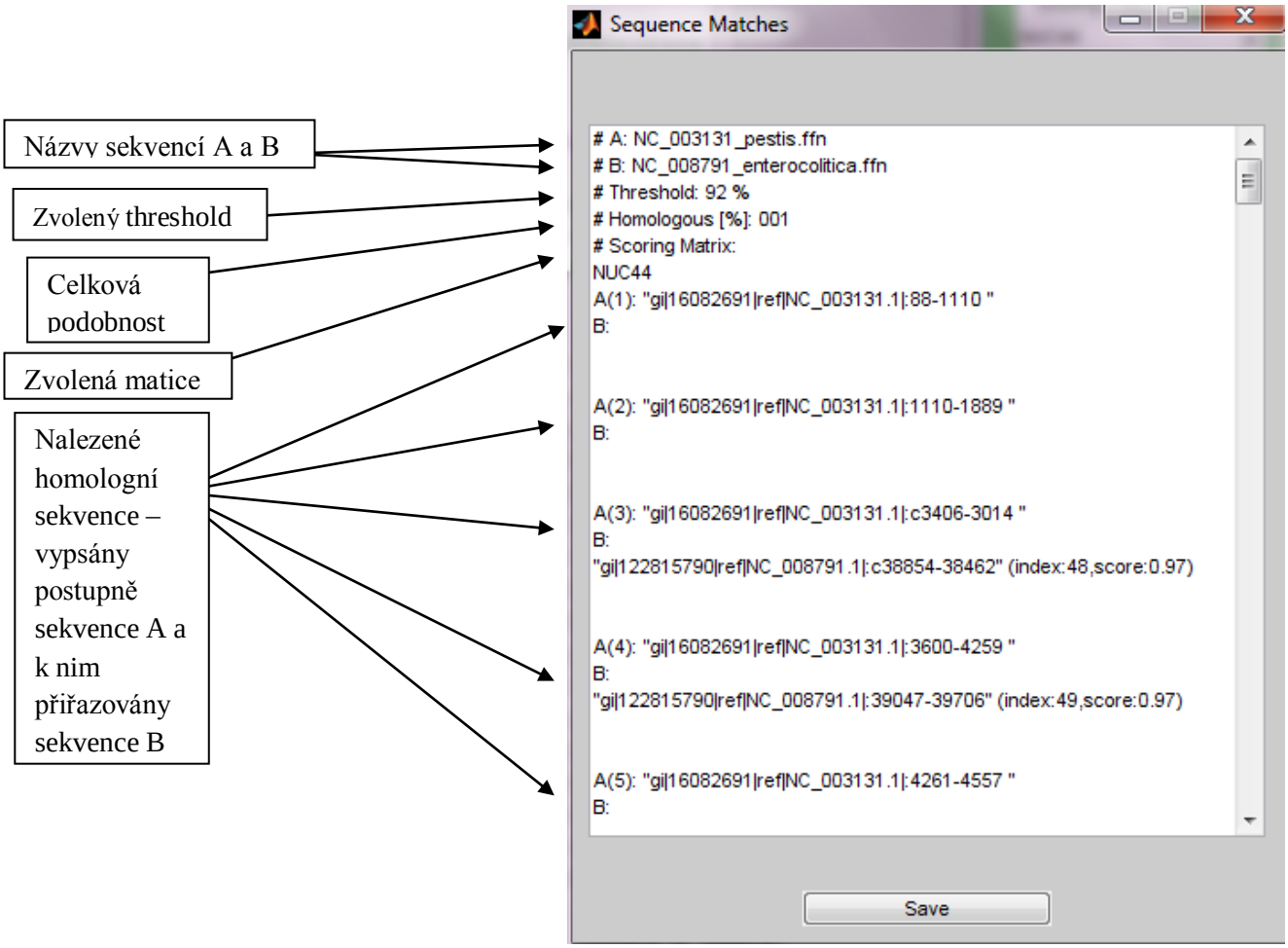
Následuje již samotné srovnání, na výběr je mezi „Compare single“ a „Compare all“.

Volba „Compare all“ srovná obě sekvence po celé jejich délce a výstupem je bodový diagram. Defaultně nastavený threshold pro shodu je nastaven na 0,50 (jako jakýsi „zlatý střed“), přičemž uživatel si může pomocí posuvníku tuto hodnotu měnit. Pod posuvníkem se uživateli v textovém boxu zobrazuje míra homologie mezi celými dvěma sekvencemi. Dále má uživatel možnost přiblížit si tu dvojici shod, která jej zrovna zajímá, a to pomocí volby „Show ij“ a následným kliknutím na onu shodu zájmu. Výsledkem je otevření nového okna, v němž je zobrazeno konkrétní zarovnání zvolené sekvence. Další možností, kterou uživatel má, je volba „Export matches“ – uživateli se otevře nové dialogové okno, v němž jsou vypsány shodné sekvence nalezené při zvolené matici, penalizaci mezer a thresholdu. Takto vypsané sekvence je možno si po volbě „Save“ uložit. Výstup z volby „Compare all“ je znázorněn na obrázku 18 a 19.

Pomocí „Compare all“ lze také provést analýzu jedné sekvence. V tomto případě se jako sekvence A i B nahraje tatáž sekvence a zvolí se „Compare all“.



Obrázek 18: Vykreslení bodového diagramu po srovnání dvou sekvencí volbou „Compare all“.



Obrázek 19: Výstup po volbě „Compare all“ – „Export Matches“ – nové dialogové okno, v němž jsou vypsány názvy sekvence, podmínky, za nichž srovnání proběhlo a především výpis nalezených homologů.

„Compare single“ je volba pro případ, že uživatele zajímá srovnání sekvence jednoho konkrétního genu. V tom případě změní v textovém poli „Index A/B“ defaultní hodnotu „1“ na takovou, jaká odpovídá pořadí genů, který jej aktuálně zajímá (což je uvedeno ve FASTA zápisu této sekvence). Po kliknutí na „Compare single“ dojde k otevření nového okna, v němž je zobrazeno zarovnání a vypsána hodnota identity.

Druhou možností, kdy použít volbu „Compare single“ je situace, kdy je bodový diagram příliš „hustě pokryt“ shodami na to, aby bylo možno zvolit srovnání mezi konkrétními dvěma geny pouze kurzorem. Postup je takový, že uživatel klasicky nahraje dvě sekvence, srovná je pomocí „Compare all“, zvolí „Export matches“ (s požadovaným thresholdem) a z takto vypsaného „katalogu“ homologních genů si vybere právě ty dva, které jej zajímají. Následně se podívá, jaký index je u nich uveden. Tento pak zapíše do polí „Index A/B“ a provede „Compare single“ dle návodu v předchozím odstavci.

5.2 Kompatibilita programu a použitý hardware

Vytvořený program je možné spustit na počítačích, které mají nainstalovanou aplikaci Matlab. Program byl vytvořen v Matlabu R2008b verze 7.7.0 a otestován byl verzí Matlabu 8.1 2013a. S oběma verzemi je program plně kompatibilní.

Program byl testován na třech počítačích – jednom přenosném a dvou stolních. Konfigurace použitých počítačů je uvedena v tabulce 4.

Tabulka 4: Konfigurace počítačů použitých pro testování programu.

Konfigurace testovacích počítačů			
Typ počítače	Přenosný počítač	Stolní počítač	Stolní počítač
Typ procesoru	Intel Core i3 M380	Intel Core 2 Quad q6600	Intel Core i2700k
Počet jader	2	4	4 (8 HT)
Takt procesoru	2,53 GHz	2,4 GHz	4,4 GHz
RAM	4 GB	4 GB	16 GB
Operační systém	Windows 7 Home Premium 64 bit	Windows 7 Professional 64 bit	Windows 7 Professional 64 bit
Verze /Matlabu	Matlab 7.7.0 R2008b	Matlab 7.7.0 R2008b	Matlab 8.1 R2013a

Rychlost, s jakou program provede srovnání, se odvíjí od výkonnosti hardwaru, na kterém program běží. Na stolním počítači s vyšším taktem jádra probíhal výpočet výrazně rychleji než například na přenosném s nižším taktem jádra.

Dalším faktorem, který hraje roli při rychlosti výpočtu, je počet jader procesoru. Zde je ovšem nutno zmínit fakt, že kód je psán bez použití Parallel Computing Toolboxu, a tedy je při výpočtu zapojeno pouze jedno jádro bez ohledu na to, kolika jádry procesor disponuje. Za situace, že by kód byl přepsán v Parallel Computing Toolboxu, by výpočet mohl probíhat při zapojení osmi jader až 8x rychleji než při zapojení jednoho jádra. Přepis pomocí Parallel Computing Toolboxu byl bohužel neúspěšný, jelikož ačkoliv implementace jako taková úspěšná byla, procesor navzdory tomu nevykazoval zvýšení výkonu. Pro úspěšné použití multijádrového výpočtu by byla nezbytná další detailnější optimalizace kódu, ke které již nemohlo dojít, jelikož k používání Parallel Computing Toolboxu bylo přistoupeno až ve chvíli, kdy byl celý kód kompletně hotov, a tedy by muselo dojít k přepsání celého skriptu vzhledem k povaze proměnných.

Ovšem i v případě úspěšného použití Parallel Computing Toolboxu by výpočet byl zrychlen pouze tolikrát, kolikrát více jader je zapojeno do výpočtu.

Je nutno zohlednit, že složitost, a tedy i doba výpočtu, roste kvadraticky se zvětšením vložené sekvence (pracujeme v podstatě s maticí o délce N, v níž se nachází N^2 položek). V tomto případě se tedy jedná o kvadratickou asymptotickou složitost k vyjádření efektivity a rychlosti vykonání algoritmu. Pokud tedy vložíme sekvenci 10x větší než sekvence předchozí, doba výpočtu se znásobí 100x. Sekvence v rozsahu stovek bází je schopen program porovnat bez problémů v relativně krátkém časovém úseku (rozmezí minut – 2 hodiny), avšak pokud bychom chtěli srovnávat sekvence velké tisíce bází, výpočet by se velice výrazně prodloužil.

6 Výsledky

Funkčnost programu byla ověřena na dvojicích sekvencí, a to jak pro sekvence nukleotidů, tak aminokyselin. V každé testované dvojici byly použity různé vstupní podmínky zarovnání (volba substituční matice, hodnota za penalizaci mezer). Následně bylo zhodnoceno, které podmínky nalézají homologie nejlépe, což bylo konfrontováno s komerčně dostupným programem. Za homologní lze považovat sekvence, u nichž byla dle programu nalezena shoda nad 35% (viz kapitola 1.4).

Vzhledem k faktu, že naprostá většina komerčně dostupných programů je založena na zarovnávání lokálním, eventuálně mnohočetném, byl výběr programu, který by pracoval na mechanismu totožném jako mnou napsaný program (a ještě umožňoval změny ve volbě vstupních parametrů) velice zúžen.

Jako referenční programy (pro ověření správnosti zarovnání, a tedy nalézání homologů) byly použity následující dva:

1) EMBOSS Needle/Stretcher – Pairwise Sequence Alignment – provozován EMBL – EBI. Využívá globální zarovnání a při vkládání sekvencí je také možno zvolit si substituční matici a penalizaci vnesení mezer. Tento program byl použit jako kontrolní pro zarovnání nukleotidů. Dostupný je na této internetové adrese: http://www.ebi.ac.uk/Tools/services/web_emboss_needle/toolform.ebi.
2) FASTA Sequence Comparison – konkrétně nástroj ggsearch. Využívá opět globálního zarovnání a je zde také možné zvolit substituční matici. Byl použit jako kontrolní program pro zarovnání proteinů. Je dostupný na této internetové adrese: http://fasta.bioch.virginia.edu/fasta_www2/fasta_www.cgi.

K porovnání výsledků nalezených homologií mým programem s nějakou komerčně dostupnou databází homologií byl použit PSAT (Prokaryotic Sequence Analysis Tool) – dostupný z těchto internetových stránek: http://nwrce.org/cgi-bin/psat/select.cgi. Toto porovnání skýtá však dva problémy. Prvním problémem je fakt, že nejsou dostupné informace o tom, jaký algoritmus či snad dokonce jaké vstupní parametry matice a hodnoty penalizací mezer program používá (avšak vzhledem k zřídkakdy se vyskytujícímu komerčnímu využití globálního zarovnání se dá předpokládat, že program pracuje na jiném algoritmu, než program můj). Druhým problémem je fakt, že veškeré referenční databáze (a tedy pochopitelně i PSAT) umožňují pouze srovnání velmi velkých sekvencí (vždy jedna ze sekvencí je kompletní , v databázi uložený genom , a druhou je dotazovaná sekvence) a mnou vytvořený programem by takto velké sekvence srovnával klidně několik týdnů.

Toto „velikostní“ omezení pro vkládané sekvence také velice úzce souvisí s výběrem testovacích dvojic sekvencí. Sekvence byly vybírány především na základě délky (a ne tedy na základě zajímavých biologických relací), která by byla dostatečně dlouhá pro relevantní zarovnání (minimum je 10 bp) a zároveň také „přiměřeně dlouhá“ vzhledem k době trvání

výpočtu (např. srovnání sekvencí o velikosti .ffn souboru 3000 kB krát 3000 kB trvalo přibližně 12 hodin).

Názvy jednotlivých sekvencí jsou vždy určeny podle databáze UniProt či NCBI.

6.1 Porovnání správnosti zarovnání s programy EMBOSS a FASTA

V této části analýzy výsledků bylo testováno, zdali mnou vytvořený program správně zarovnává sekvence. Dále bylo testováno, jaké je nejvhodnější nastavení vstupních parametrů – jaká matice je nejlepší, jaké je nejvhodnější penalizování mezer a především to, zdali je k určování homologií vhodnější vkládat sekvence nukleotidů či aminokyselin.

6.1.1 *Acinetobacter baumannii BJAB0715* vs. *Acinetobacter baumannii BJAB0868*

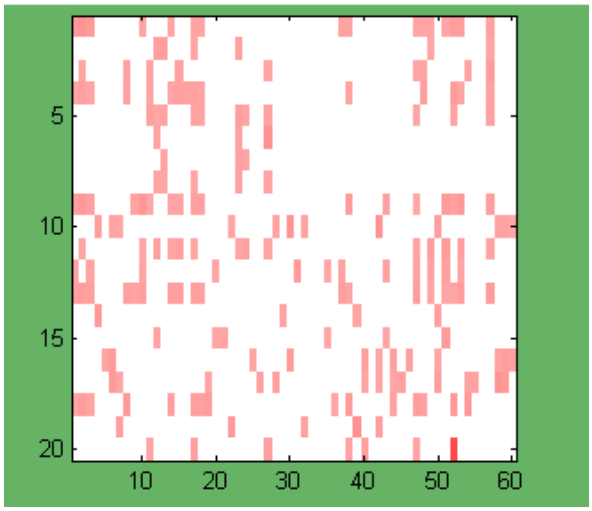
V tomto srovnání je první sekvence pocházející z *Acinetobacter baumannii BJAB0715*, konkrétně se jedná o 50S ribozomální protein L18. V této sekvence se nachází celkem 60 geny kódujících úseků. Druhou sekvencí je *Acinetobacter baumannii BJAB0868*, konkrétně jde o enzym isocitrátovou lyázu. Druhá sekvence kóduje 20 genů.

Nastavené vstupní parametry:

1) Srovnání nukleotidové sekvence, matice: nuc44, penalizace otvírací mezery -10, rozšiřovací mezery -0,5, threshold 35%.
2) Srovnání nukleotidové sekvence, matice: nuc44, penalizace otvírací mezery -10, rozšiřovací mezery -10, threshold 35%.
3) Srovnání nukleotidové sekvence, matice: identity, penalizace otvírací mezery -10, rozšiřovací mezery -0,5, threshold 35%.
4) Srovnání nukleotidové sekvence, matice: tranzitně transverzní, penalizace otvírací mezery -10, rozšiřovací mezery -0,5, threshold 35%.
5) Srovnání proteinové sekvence, matice: BLOSUM50, penalizace otvírací mezery -10, rozšiřovací mezery -2, threshold 35%.
6) Srovnání proteinové sekvence, matice: PAM250, penalizace otvírací mezery -10, rozšiřovací mezery -2, threshold 35%.

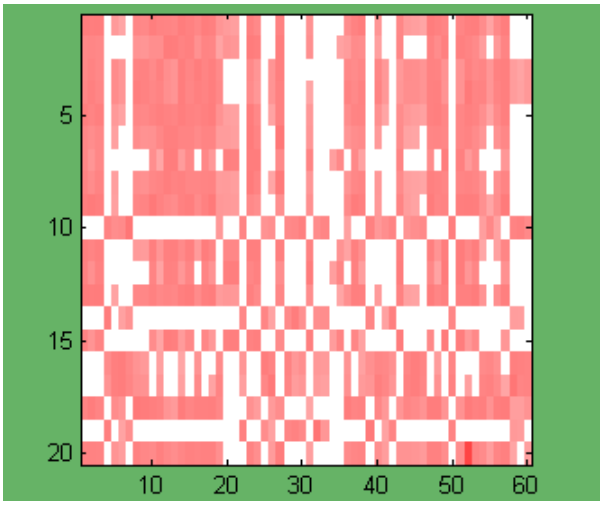
Výsledky:

1) Celková podobnost sekvencí – 1,7%. Nalezené homologní sekvence viz obrázek 20.



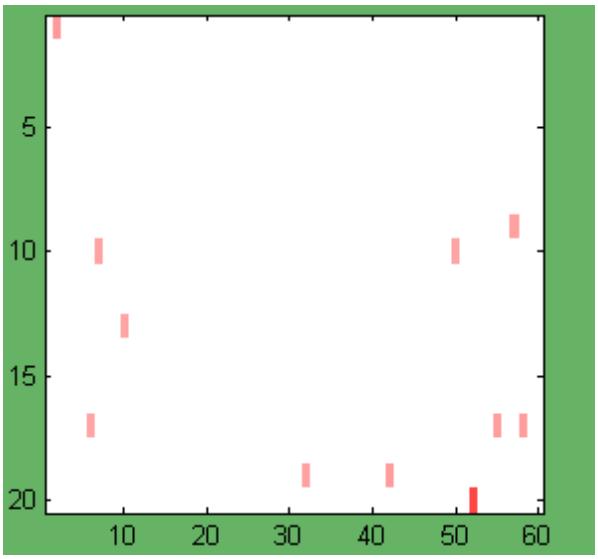
Obrázek 20: Nalezené homologní sekvence u *Acinetobacter baumannii BJAB0715* a *Acinetobacter baumannii BJAB0868* dle nastavených parametrů 1).

2) Celková podobnost sekvencí – 5,9%. Nalezené homologní sekvence viz obrázek 21.



Obrázek 21: Nalezené homologní sekvence u *Acinetobacter baumannii BJAB0715* a *Acinetobacter baumannii BJAB0868* dle nastavených parametrů 2).

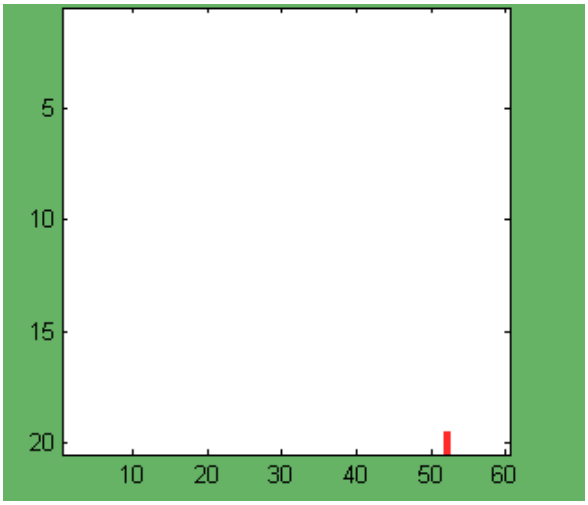
3) Celková podobnost sekvencí – 0,1%. Nalezené homologní sekvence viz obrázek 22.



Obrázek 22: Nalezené homologní sekvence u *Acinetobacter baumannii BJAB0715* a *Acinetobacter baumannii BJAB0868* dle nastavených parametrů 3).

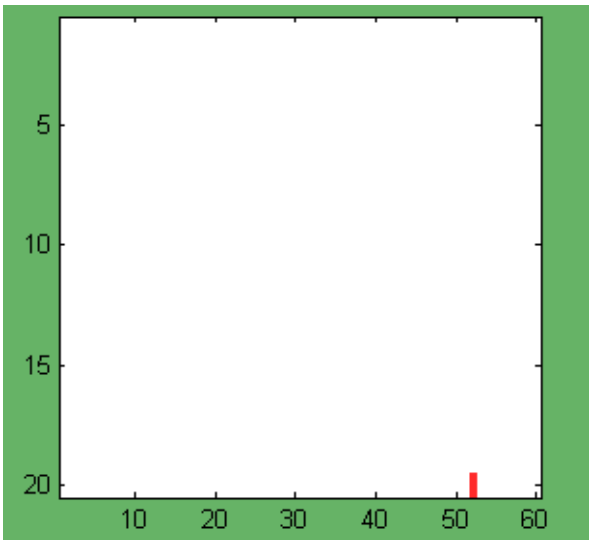
4) Celková podobnost sekvencí – 0,0%. Nenalezeny žádné homologní sekvence.

5) Celková podobnost sekvencí – 0,0% (zde je třeba podotknout, že zaokrouhlování je nastaveno na dvě desetinná místa). Nalezené homologní sekvence viz obrázek 23.



Obrázek 23: Nalezené homologní sekvence u *Acinetobacter baumannii BJAB0715* a *Acinetobacter baumannii BJAB0868* dle nastavených parametrů 5).

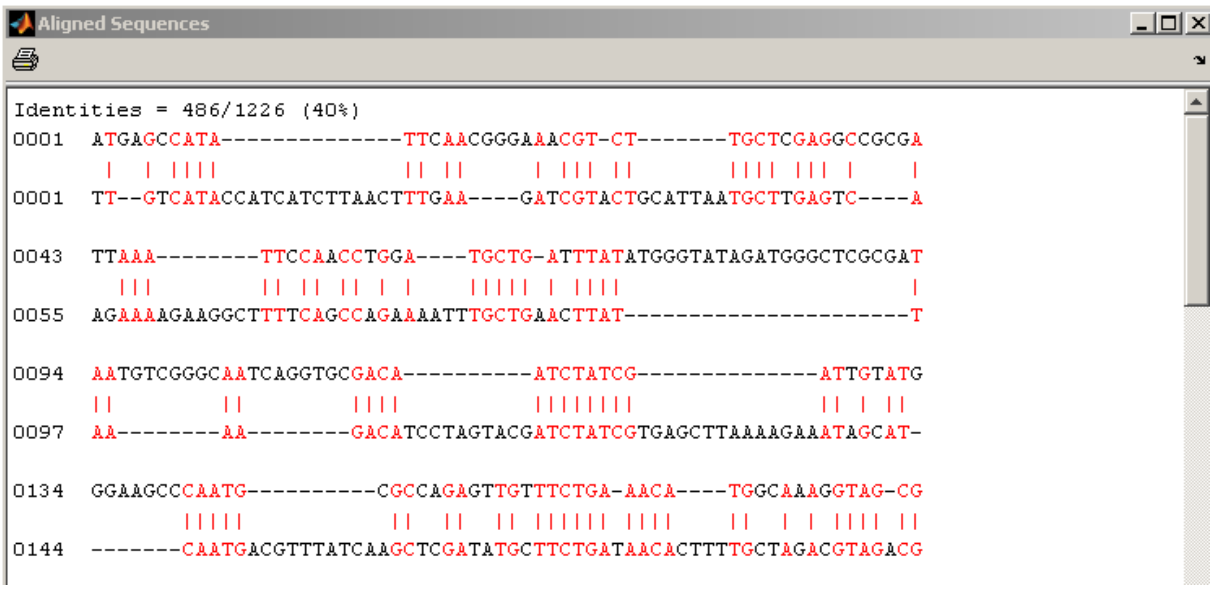
6) Celková podobnost sekvencí – 0,0%. Nalezené homologie viz obrázek 24.



Obrázek 24: Nalezené homologní sekvence u *Acinetobacter baumannii BJAB0715* a *Acinetobacter baumannii BJAB0868* dle nastavených parametrů 6).

Dle testovací databáze EMBOSS Needle byla ověřena správnost zarovnání v případě vložení nukleotidových sekvencí. Vstupní parametry EMBOSS Needle byly zcela identické jako nastavené vstupní parametry 1). Zarovnání mnou vytvořeným programem a programem EMBOSS Needle se lišil ve výsledku v řádech desetin.

Příkladem shody zarovnání je zarovnání vždy prvního genu z každé sekvence. Zarovnání pomocí mého programu je znázorněno na obrázku 25 a zarovnání pomocí EMBOSS Needle na obrázku 26.

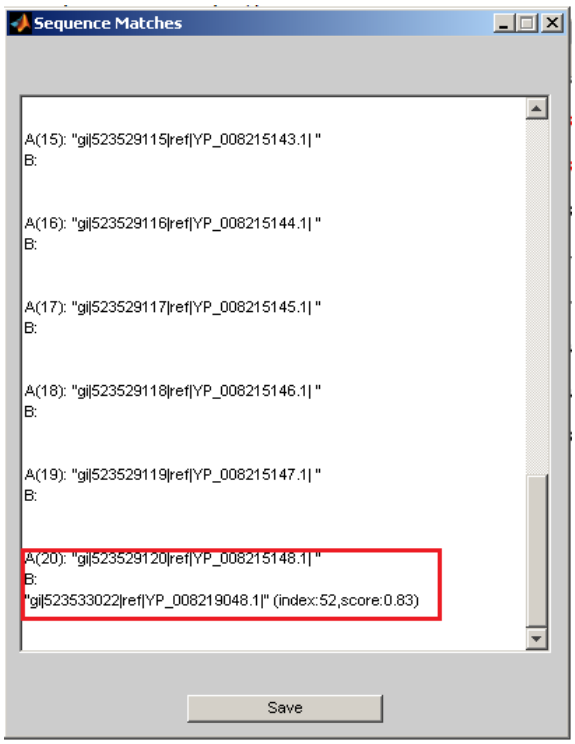


Obrázek 25: Výsledek zarovnání sekvencí prvního genu *Acinetobacter baumannii BJAB0715* a *Acinetobacter baumannii BJAB0868* mnou vytvořeným programem.

```
# Length: 1199
# Identity:     485/1199 (40.5%)
# Similarity:   485/1199 (40.5%)
# Gaps:         571/1199 (47.6%)
# Score: 762.5
#
#
#=======================================

EMBOSS_001         1 TTGTCATACCAT----CATCTT-AACTTTGAAGATCGTACTGCATTAATG     45
                           ||    |||.|| |||   |...|.||| ||       ||
EMBOSS_001         1 ----------ATGAGCCATATTCAAC---GGGAAACGT-CT-------TG     29

EMBOSS_001        46 CTTGAGTC----AAGAAAAGAAGGCTTTTCAGCCAGAAAATTTGCTGAAC     91
                     ||.|||.|    |..|||        ||.||.||.|.|    ||||| |.
EMBOSS_001        30 CTCGAGGCCGCGATTAAA--------TTCCAACCTGGA----TGCTG-AT     66

EMBOSS_001        92 TTAT-----TA----------------AAAGAC-----ATCCTAG----T    111
                     ||||     ||                ||.|.|     |||  ||    .
EMBOSS_001        67 TTATATGGGTATAGATGGGCTCGCGATAATGTCGGGCAATC--AGGTGCG    114

EMBOSS_001       112 ACGATCTATCGTGAGCTTAAAAGAAATAGCAT--------CAATGACGTT    153
                     ||.||||||||              ||.|.||        |||||
EMBOSS_001       115 ACAATCTATCG--------------ATTGTATGGGAAGCCCAATG-----    145
```

Obrázek 26: Zarovnání sekvencí prvního genu *Acinetobacter baumannii BJAB0715* a *Acinetobacter baumannii BJAB0868* programem EMBOSS Needle.

Ověření správnosti zarovnání proteinových sekvencí pomocí nástroje ggsearch (FASTA SEQUENCE COMPARISON) – vstupní parametry byly identické jako nastavené vstupní parametry mého programu v případě 5). Z obrázku je patrno, že byla nalezena pouze jedna jediná homologie, což prokázalo i srovnání nástrojem ggsearch. Hodnota nalezené identity se opět liší v řádu desetin procenta (nástroj ggsearch – 83,3% identity, můj program – 83%).

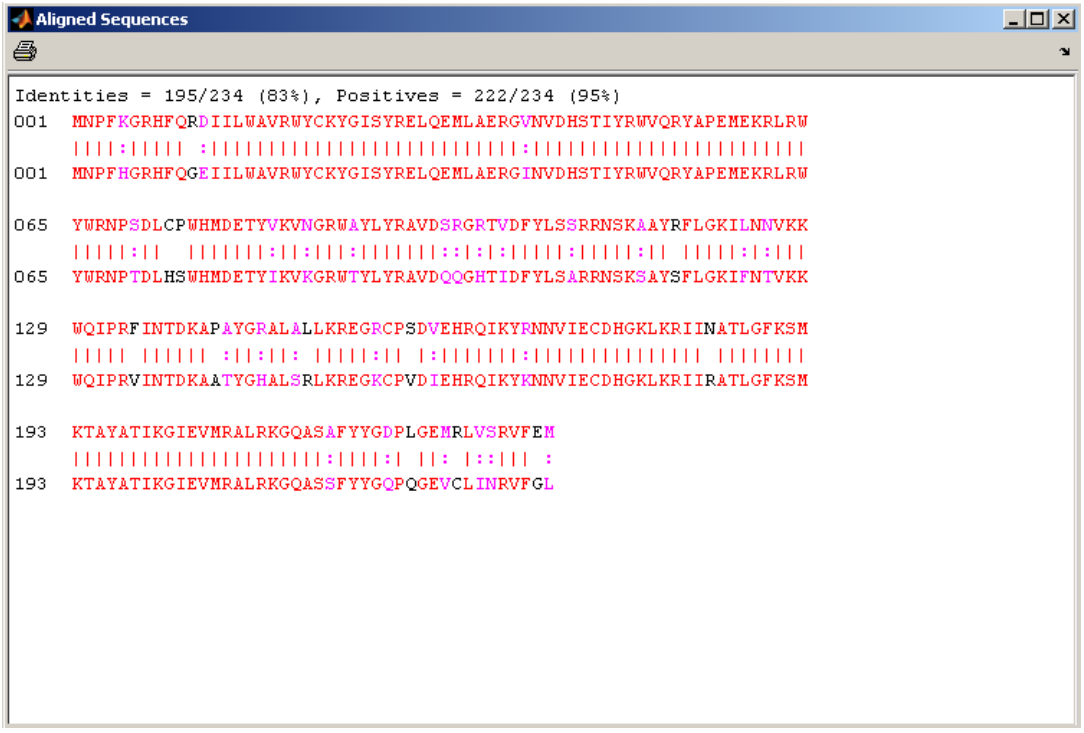


Obrázek 27: Nalezená shoda (jedna jediná) při volbě parametrů 5) – zobrazena po zvolení „Export Matches“.

```
>>>gi|523529120|ref|YP_008215148.1|, 234 aa vs TMP.q2 library

>>ref|YP_008219048.1| Transposase-like protein [Acinetobacter b              (234 aa)
 n-w opt: 1162  Z-score: 585.9  bits: 115.7 E(10000):    0
global/global (N-W) score: 1162; 83.3% identity (96.6% similar) in 234 aa overlap (1-234:1-234)
Sequence Lookup  Re-search database  General re-search
```


```
                                                                    [Domains]

[alignment]
               10        20        30        40        50        60        70        80
ref|Y  MNPFKGRHFQRDIILWAVRWYCKYGISYRELQEMLAERGVNVDHSTIYRWVQRYAPEMEKRLRWYWRNPSDLCPWHMDET
       ::::.::::: .:::::::::::::::::::::::::.:::::::::::::::::::::::::::::.:: .:::::::
ref|YP MNPFHGRHFQGEIILWAVRWYCKYGISYRELQEMLAERGINVDHSTIYRWVQRYAPEMEKRLRWYWRNPTDLHSWHMDET
               10        20        30        40        50        60        70        80

               90       100       110       120       130       140       150       160
ref|Y  YVKVNGRWAYLYRAVDSRGRTVDFYLSSRRNSKAAYRFLGKILNNVKKWQIPRFINTDKAPAYGRALALLKREGRCPSDV
       :.::.:::.:::::::: .:.:.:::::.:::::.::.::::::.:.::::::::: :::::::.::.::. ::::::.:: :.
ref|YP YIKVKGRWTYLYRAVDQQGHTIDFYLSARRNSKSAYSFLGKIFNTVKKWQIPRVINTDKAATYGHALSRLKREGKCPVDI
               90       100       110       120       130       140       150       160

              170       180       190       200       210       220       230
ref|Y  EHRQIKYRNNVIECDHGKLKRIINATLGFKSMKTAYATIKGIEVMRALRKGQASAFYYGDPLGEMRLVSRVFEM
       :::::::.:::::::::::::::.:::::::::::::::::::::::::::::::.::::.: ::. :..:::..
ref|YP EHRQIKYKNNVIECDHGKLKRIIRATLGFKSMKTAYATIKGIEVMRALRKGQASSFYYGQPQGEVCLINRVFGL
              170       180       190       200       210       220       230
```

Obrázek 28: Nalezená shoda při volbě parametrů 5) – zarovnání *Acinetobacter baumannii BJAB0715* a *Acinetobacter baumannii BJAB0868* pomocí nástroje ggsearch.



Obrázek 29: Nalezená shoda při volbě parametrů 5) – zarovnání *Acinetobacter baumannii BJAB0715* a *Acinetobacter baumannii BJAB0868* mnou vytvořeným programem.

Z obrázků 27 a 28 je patrno, že mnou vytvořený program a komerčně dostupný nástroj ggserach nalezly homologii na skutečně identickém místě, a to se skoro stejnou přesností.

Celkové shrnutí: V tomto srovnání se ukázalo být vhodnějším použití srovnání dvou sekvencí aminokyselin. Lze také poměrně spolehlivě vyvodit závěr, že gen na pozici 20 v sekvenci A a gen na pozici 52 v sekvenci B jsou homologní.

6.1.2 *Salmonella enterica subsp. enterica serovar Typhimurium SL1344* vs. *Staphylococcus aureus uid193761*

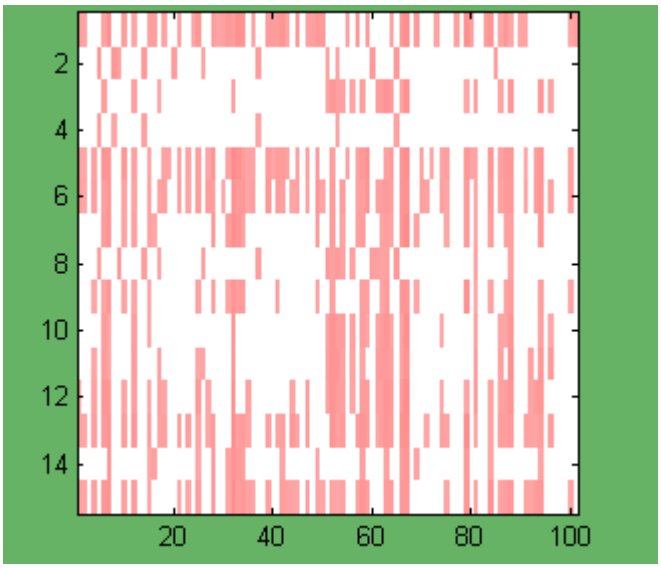
První sekvencí je sekvence G- tyčinky *Salmonella enterica subsp. enterica* serovar *Typhimurium SL1344*, konkrétně jde řetězec její cirkulární DNA, v této sekvenci je celkem 101 kódujících úseků. Druhá sekvence pochází z G+koka *Staphylococcus aureus uid193761*, konkrétně malá část jeho genomu o velikost 15 kódujících úseků. Na této dvojici je testován vliv penalizace mezer na finální zarovnání.

Nastavené vstupní parametry:

1) Srovnání nukleotidové sekvence, matice: nuc44, penalizace otvírací mezery -10, rozšiřovací mezery -0,5, threshold 35%.
2) Srovnání nukleotidové sekvence, matice: nuc44, penalizace otvírací mezery -10, rozšiřovací mezery -10, threshold 35%.
3) Srovnání nukleotidové sekvence, matice: nuc44, penalizace otvírací mezery -1, rozšiřovací mezery -1, threshold 35%.
4) Srovnání nukleotidové sekvence, matice: nuc44, penalizace otvírací mezery -20, rozšiřovací mezery -1, threshold 35%.

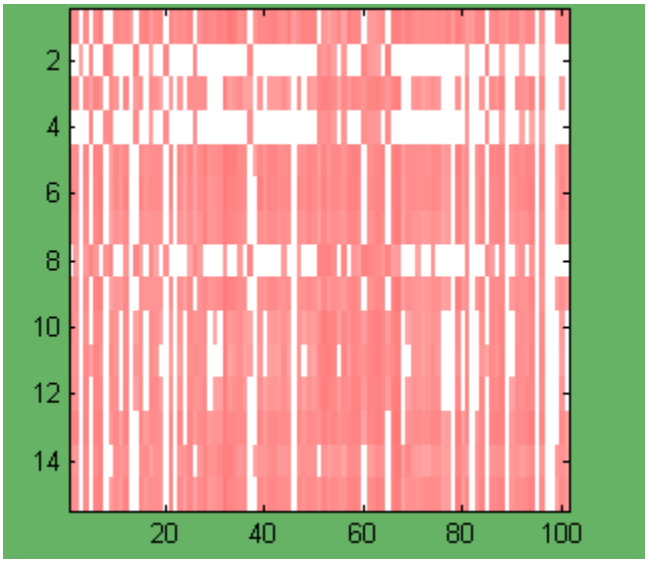
Výsledky:

1) Celková podobnost sekvencí – 3,2%. Nalezené homologie viz obrázek 30.



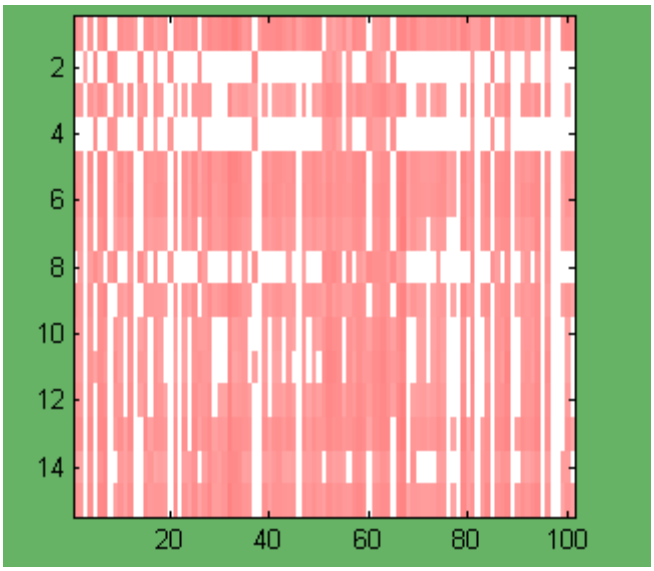
Obrázek 30: Nalezené homologní sekvence u *Salmonella enterica subsp. enterica serovar Typhimurium SL1344* a *Staphylococcus aureus uid193761* dle nastavených parametrů 1).

2) Celková podobnost sekvencí – 6,8%. Nalezené homologie viz obrázek 31.



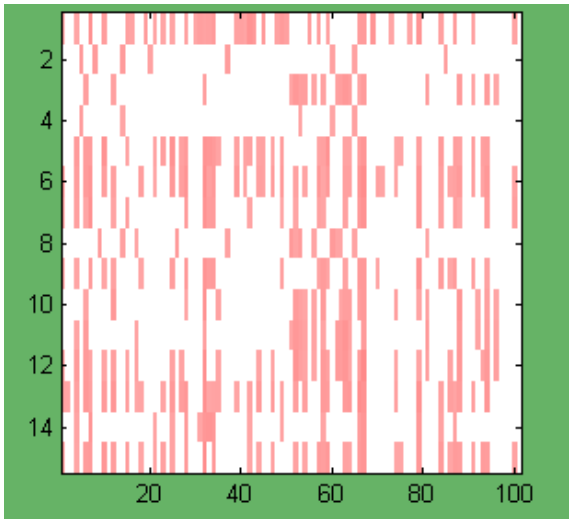
Obrázek 31: Nalezené homologní sekvence u *Salmonella enterica subsp. enterica serovar Typhimurium SL1344* a *Staphylococcus aureus uid193761* dle nastavených parametrů 2).

3) Celková podobnost sekvencí – 6,5%. Nalezené homologie viz obrázek 32.



Obrázek 32: Nalezené homologní sekvence u *Salmonella enterica subsp. enterica serovar Typhimurium SL1344* a *Staphylococcus aureus uid193761* dle nastavených parametrů 3).

4) Celková podobnost – 2,5%. Nalezené homologie viz obrázek 33.



Obrázek 33: Nalezené homologní sekvence u *Salmonella enterica subsp. enterica serovar Typhimurium SL1344* a *Staphylococcus aureus uid193761* dle nastavených parametrů 4).

Celkové shrnutí: Na této dvojici vzorků byl testován vliv penalizace mezer na výsledek zarovnání. Experimentálně ověřené hodnoty penalizace (a také hodnoty používané v komerčně dostupných programech jsou pro otvírací mezeru -10 a pro rozšiřovací mezeru -0,5. Tyto hodnoty odpovídají nastavení vstupních parametrů 1). Z nastavení parametrů 2) a 3) je patrno, že pokud bude ohodnocena stejně jak otvírací mezera, tak rozšiřovací, zarovnání bude nepřesné a „nadhodnocené“, a tedy neoptimální. Nastavení vstupních parametrů 4) se hodnotami nejvíce blíží hodnotám pro optimální zarovnání, ale je zde patrno, že díky příliš

velkému rozdílu ohodnocení mezi otvírací a rozšiřovací mezerou, je zarovnání naopak „podhodnoceno“, a tedy opět není optimální. Správnost těchto srovnání byla opět potvrzena konfrontací s výsledky, které poskytl program EMBOSS Needle.

Pozice nalezených homologických úseků tedy nejlépe odpovídají těm, které jsou uvedeny na obrázku 30.

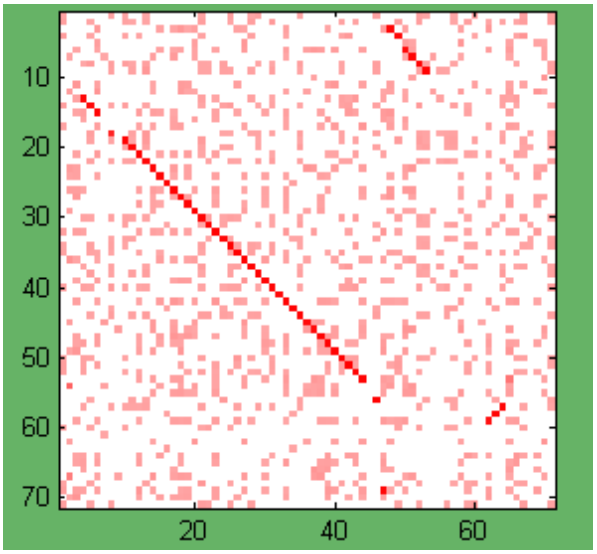
6.1.3 *Yersinia enterocolitica subsp. enterocolitica 8081* vs. *Yersinia pestis CO92* plasmid pCD1

Třetí testovací dvojicí sekvencí jsou dva různé druhy rodu *Yersinia* - *Yersinia enterocolitica subsp. enterocolitica* 8081a *Yersinia pestis* CO92 plasmid pCD1. Velikost obou sekvencí je shodně 71 geny kódujících úseků.

Nastavené vstupní parametry:

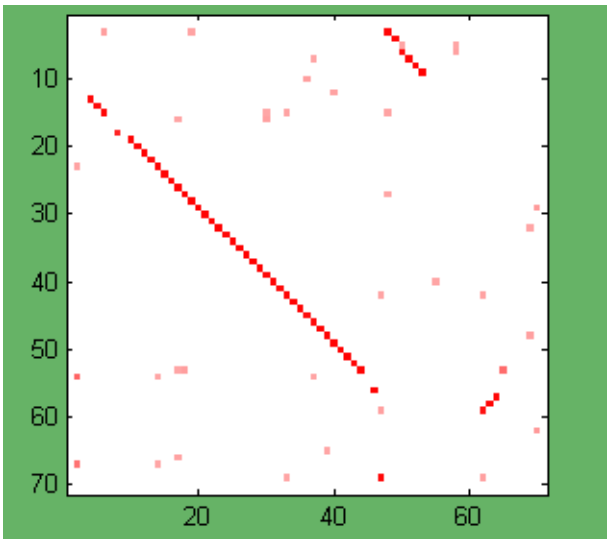
1) Srovnání nukleotidové sekvence, matice: nuc44, penalizace otvírací mezery -10, rozšiřovací mezery -0,5, threshold 35%.
2) Srovnání nukleotidové sekvence, matice: identity, penalizace otvírací mezery -10, rozšiřovací mezery -0,5, threshold 35%.
3) Srovnání proteinové sekvence, matice: BLOSUM62, penalizace otvírací mezery -10, rozšiřovací mezery -0,5, threshold 35%.
4) Srovnání proteinové sekvence, matice: PAM250, penalizace otvírací mezery -10, rozšiřovací mezery -0,5, threshold 35%.

1) Celková podobnost – 2,2%. Nalezené homologie viz obrázek 34.



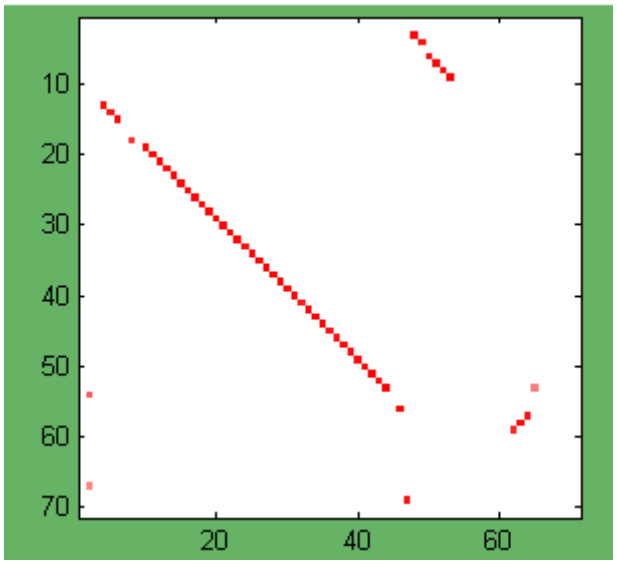
Obrázek 34: Nalezené homologní sekvence u *Yersinia enterocolitica subsp. enterocolitica* 8081 a *Yersinia pestis* CO92 plasmid pCD1 dle nastavených parametrů 1).

2) Celková podobnost – 0,2%. Nalezené homologie viz obrázek 35.



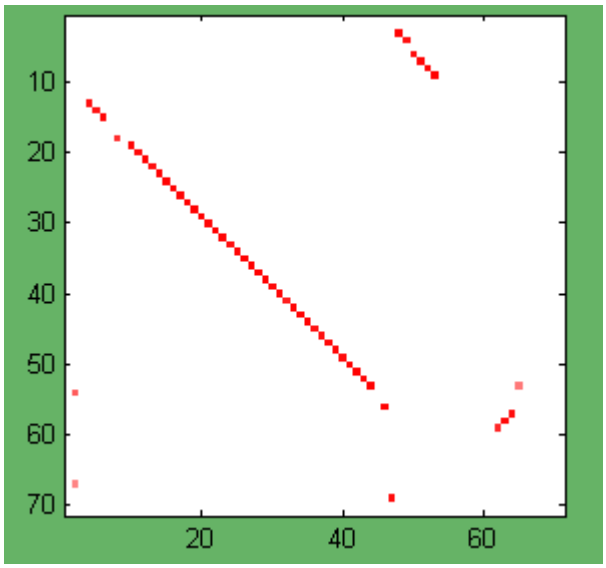
Obrázek 35: Nalezené homologní sekvence u *Yersinia enterocolitica subsp. enterocolitica* 8081 a *Yersinia pestis* CO92 plasmid pCD1 dle nastavených parametrů 2).

3) Celková podobnost – 0,1%. Nalezení homologie viz obrázek 36.



Obrázek 36: Nalezené homologní sekvence u *Yersinia enterocolitica subsp. enterocolitica* 8081 a *Yersinia pestis* CO92 plasmid pCD1 dle nastavených parametrů 3).

4) Celková podobnost – 0,1%. Nalezené homologie viz obrázek 37.

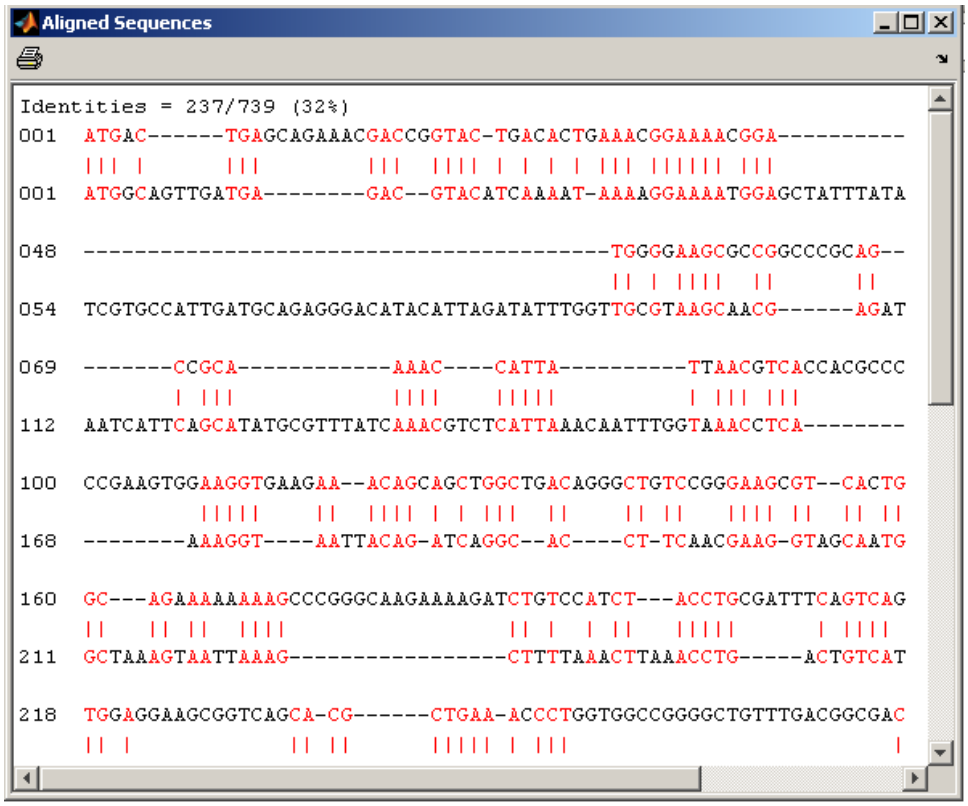


Obrázek 37: Nalezené homologní sekvence u *Yersinia enterocolitica subsp. enterocolitica* 8081 a *Yersinia pestis* CO92 plasmid pCD1 dle nastavených parametrů 4).

Na obrázcích 38 a 39 je uvedeno porovnání výsledků zarovnání programem EMBOSS Needle a mnou vytvořeným programem (pro názornost – vzhledem k velikosti výstupního souboru – je opět uvedeno srovnání sekvencí prvního genu) při zvolených parametrech 1):

```
..
# Length: 684
# Identity:     254/684 (37.1%)
# Similarity:   254/684 (37.1%)
# Gaps:         345/684 (50.4%)
# Score: 307.0
#
#
#=======================================

EMBOSS_001         1 ATGAC---TGAGCAGAAACGACCGGTAC-TGACACTGAAACGGAAAACGG     46
                     |||.|   |||..|||       .|||| |.|.|.| |||.||||||.||
EMBOSS_001         1 ATGGCAGTTGATGAGA-------CGTACATCAAAAT-AAAAGGAAAATGG     42

EMBOSS_001        47 ATGGGGAAGCGC---------CGGCCC------GCAGCCGCAAAACCAT-     80
                     |         ||         ||..||      ||||..|.|.|..|||
EMBOSS_001        43 A---------GCTATTTATATCGTGCCATTGATGCAGAGGGACATACATT     83

EMBOSS_001        81 ---TATT-------------AACG----------TCACC-----------     93
                        ||||             ||||          |||.|
EMBOSS_001        84 AGATATTTGGTTGCGTAAGCAACGAGATAATCATTCAGCATATGCGTTTA    133

EMBOSS_001        94 ----ACGCCCC------CGAAGTGG----------AAGGTGAAGAA--AC    121
                         |||.|.|      |.|..|||          |||||    ||  ||
EMBOSS_001       134 TCAAACGTCTCATTAAACAATTTGGTAAACCTCAAAAGGT----AATTAC    179

EMBOSS_001       122 AGCAGCTGGCTGACAGGGCTGTCCGGGAAGCGT--CACTGGC---AGAAA    166
                     || |.|.|||  ||    || ||...|||| ||  ||.||||   ||.||
```

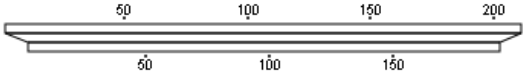
Obrázek 38: Zarovnání sekvencí prvního genu *Yersinia enterocolitica subsp. enterocolitica* 8081 a *Yersinia pestis* CO92 plasmid pCD1 programem EMBOSS Needle.



Obrázek 39: Zarovnání sekvencí prvního genu *Yersinia enterocolitica subsp. enterocolitica* 8081 a *Yersinia pestis* CO92 plasmid pCD1 mnou vytvořeným programem.

Na obrázcích 40 a 41 je uvedeno porovnání výsledků zarovnání nástrojem ggsearch programu FASTA a mnou vytvořeným programem (pro názornost – vzhledem k velikosti výstupního souboru – je opět uvedeno srovnání sekvencí jednoho nalezeného homologního genu) při zvolených parametrech 1):

```
>>>gi|386730718|ref|YP_006203694.1|, 210 aa vs TMP.q2 library

>>ref|YP_007586564.1| Resolvase/integrase [Staphylococcus aureu              (192 aa)
 n-w opt: 229  Z-score: 276.3  bits: 58.0 E(10000): 1.2e-109
global/global (N-W) score: 229; 34.4% identity (55.0% similar) in 218 aa overlap (1-210:1-192)
Sequence Lookup  Re-search database  General re-search
```

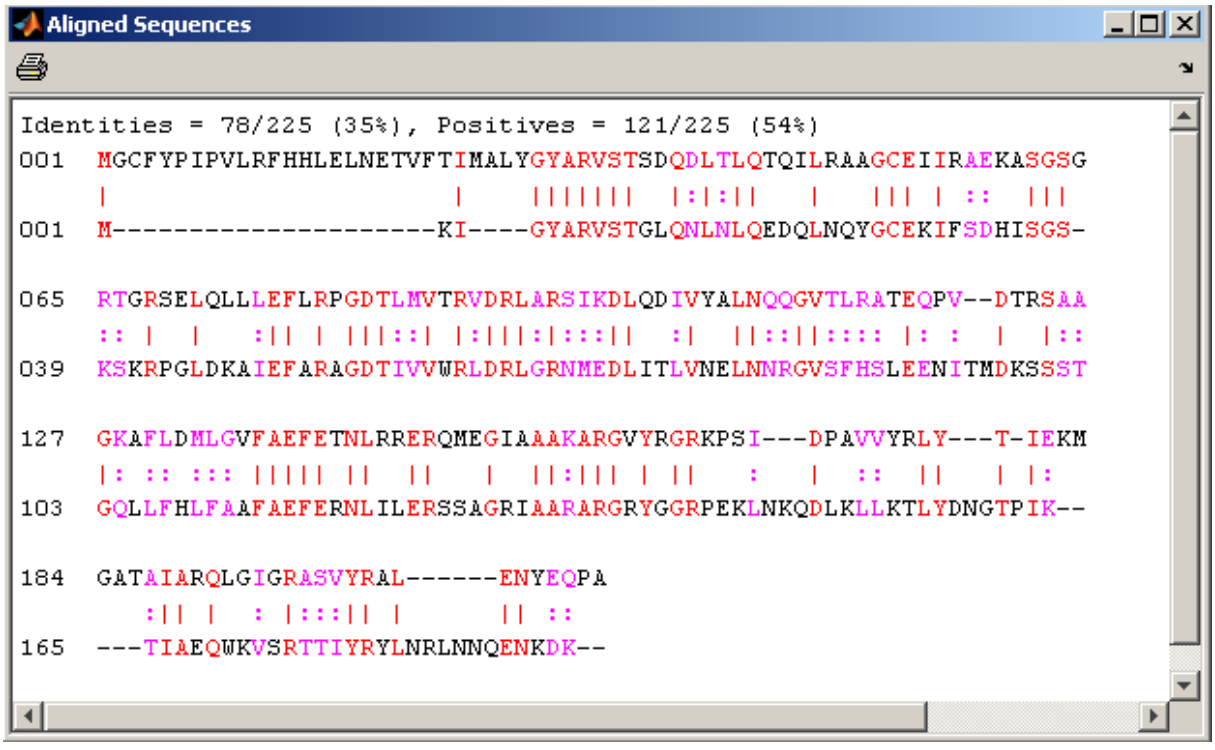


[Domains]

[alignment]

```
               10        20        30        40        50        60        70        80
ref|Y  MGCFYPIPVLRFHHLELNETVFTIMALYGYARVSTSDQDLTLQTQILRAAGCEIIRAEKASGSGRTGRSELQLLLEFLRP
       :                          :::::::  :.:.::   :   ::: : ..  ::: .. :  :   .:: :
ref|YP MKI-------------------------GYARVSTGLQNLNLQEDQLNQYGCEKIFSDHISGS-KSKRPGLDKAIEFARA
                                     10        20        30         40        50

              90       100       110       120       130       140       150
ref|Y  GDTLMVTRVDRLARSIKDLQDIVYALNQQGVTLRATEQPV--DTRSAAGKAFLDMLGVFAEFETNLRRERQMEGIAAAKA
       ::..: :.::::.:...::  .:  ::..::.... :. .  :  :..:. .. ....::::: ::  ::.  :  ::.:
ref|YP GDTIVVWRLDRLGRNMEDLITLVNELNNRGVSFHSLEENITMDKSSSTGQLLFHLFAAFAEFERNLILERSSAGRIAARA
           60        70        80        90        100       110       120       130

      160       170       180       190       200           210
ref|Y  RGVYRGRKPSIDPAVVYRLYTIEKMGA--TAIARQLGIGRASVYRALE---NYEQP-A
       :: : ::   ..   .  : :.   :.   .:: :  . :...:: :    : :.
ref|YP RGRYGGRPEKLNKQDLKLLKTLYDNGTPIKTIAEQWKVSRTTIYRYLNRLNNQENKDK
          140       150       160       170       180       190
```

Obrázek 40: Nalezená shoda při volbě parametrů 3) – zarovnání *Acinetobacter baumannii BJAB0715* a *Acinetobacter baumannii BJAB0868* pomocí nástroje ggsearch.



Obrázek 41: Nalezená shoda při volbě parametrů 3) – zarovnání *Yersinia enterocolitica subsp. enterocolitica* 8081 a *Yersinia pestis* CO92 plasmid pCD1 mnou vytvořeným programem.

Z obrázků 40 a 41 je patrno, že mnou vytvořeným program a komerčně dostupný nástroj ggserach nalezly homologii na skutečně identickém místě, a to se skoro stejnou přesností.

Celkové shrnutí: Na obrázcích 36 a 37 jsou patrny výrazně vystupující diagonály, z čehož lze vyvodit závěr, že právě tyto diagonály jsou homologními úseky sekvencí. Je nutno dodat, že tyto úseky byly dobře patrny již na obrázcích 34 a 35, avšak v těchto bodových diagramech je zobrazeno ještě mnoho dalších shod. Důvodem je fakt, že výstup na obrázcích 34 a 35 je proveden srovnáním dvou nukleotidových sekvencí, kdežto výstupy na obrázcích 36 a 37 vznikly srovnáním sekvencí aminokyselin, kde je shoda dvou pozic, vzhledem k počtu možných kombinací průkaznější, než shoda v nukleotidové sekvenci.

6.2 Porovnání nalezených homologií s databází PSAT

V této části byly porovnávány mnou nalezené homologie ve dvojicích sekvencí a homologie nalezené komerčně dostupným programe PSAT. Cílem porovnávání byl především počet nalezených homologií a poloha v sekvenci.

Ve dvou srovnávaných dvojicích je jedna ze sekvencí výrazně větší než sekvence druhá. Důvod je takový, že bylo potřeba nalézt takové dvojice, které by byl schopen v přijatelném čase srovnat mnou vytvořený program a které také obsahuje databáze PSAT. V PSAT je totiž jednou vloženou sekvencí sekvence referenční (jedná se obvykle o celý genom bakterie) a druhou sekvencí je sekvence srovnávaná/dotazovaná (takováto sekvence obsahuje už i velice krátké úseky). První (kapitola 6.2.1) a třetí (kapitola 6.2.3) testovaná dvojice tedy obsahuje jednu sekvenci, která představuje celý genom bakterie a druhou sekvencí je krátký úsek.

6.2.1 *Brucella abortus bv. 1.9-941* vs. *Legionella pneumophila* sérotyp *Paris*

Sekvence *Brucelly abortus bv. 1.9-941* nese informaci o kódování 2029 genových úseků. Oproti tomu sekvence *Legionella pneumophilla* sérotyp *Paris* je sekvencí výrazně kratší – kóduje 139 genových úseků.

Zvolené parametry v mém programu byly následující: při prvním zarovnávání matice PAM250 a při druhém zarovnávání matice BLOSUM62, otvírací mezera -10, rozšiřovací mezera -1. Zvolené parametry odpovídají těm parametrům, které byly v kapitole 6.1

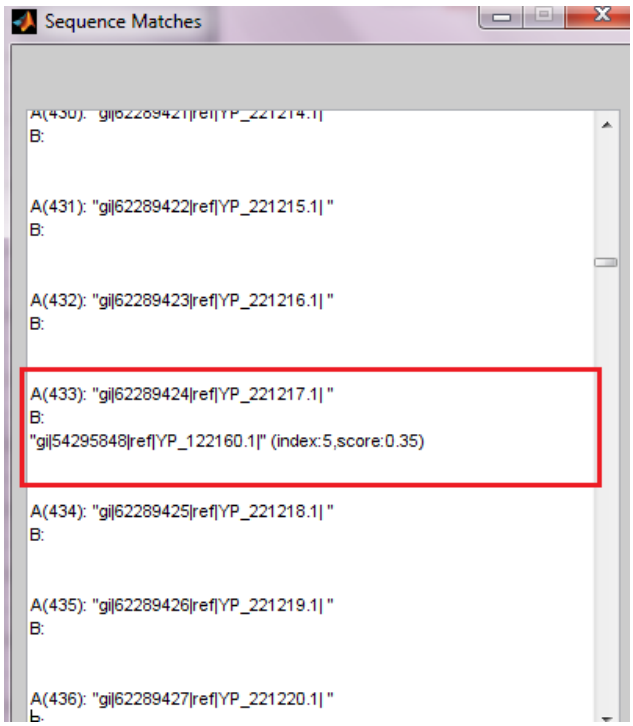
vyhodnoceny jako nejvhodnější. Srovnávaná je sekvence aminokyselin – opět dle výsledků z kapitoly 6.1.

Z defaultního nastavení programu PSAT bohužel není zjistitelná matice, penalizace mezer, ani algoritmus, kterým je zarovnáváno. Jediné volitelné hodnoty jsou E-value, bit score (jiný název pro Z-score – viz kapitola 4.1.1) a % identity. Při tomto srovnávání bylo ponecháno původní nastavení, tj. E-value < 0,1; bit score > 200; % identity > 30. Další možnou volbou je nalezení homologů v dotazované sekvenci pouze pro určitý gen nebo vypsání homologů do tabulky atd.

Výsledky:

S thresholdem 30% (stejný jako u PSAT) a s výše zmíněnými vstupními parametry mnou vytvořený program nalezl s použitím matice PAM 250 6 homologií a s použitím matice BLOSUM62 10 homologií. Při použití matice BLOSUM62 byla homologie s nejvyšším procentem identity (35%) na pozici 433 v sekvenci A a na pozici 5 v sekvenci B. Tato shoda je zobrazena na obrázku 42.

Program PSAT nalezl celkem 29 homologních sekvencí. Mezi těmito nalezenými homologiemi byla i homologie na pozici 433A a 5B. Míra identity byla určena jako 39,69%. Tato homologie je znázorněna na obrázku 43 (obrázek je uveden pouze pro názornost, jelikož výsledky, které poskytuje program PSAT jsou příliš velké a v textovém editoru obtížně reprodukovatelné).



Obrázek 42: Shoda s nejvyšším procentem identity při srovnání *Brucella abortus bv. 1.9-941* a *Legionella pneumophila* sérotyp *Paris.*

BLAST HITS (displaying first 25 hits - ordered by homolog clustering score) [Edit Settings]
Total number of hits: 1
Number of genomes with hits: 1
Meeting following criteria: e-value < 0.1; bit score > 200; % identity > 30

BLAST HIT SCORES
Click on a score heading to sort results

HOMOLOG CLUSTERING SCORE
Showing scores>=2

Genome	Genomic Element	Gene #	Start	End	Strand	Protein length	Gene Name	Locus tag	Product	% identity	Align length	e-value	Bit score	score
Brucella_abortus_bv__1_9_941	chromosome II	396	423688	424773	+	361	-	BruAb2_0423	ABC transporter, ATP-binding protein					
Legionella_pneumophila_Paris	chromosome	1115	1268818	1269912	-	364	potA	lpp1143	putrescine/spermidine ABC transporter ATPase protein	39.69	325	6e-64	237	3

Obrázek 43: Programem PSAT nalezená homologie mezi stejnými úseky *Brucella abortus bv. 1.9-941* a *Legionella pneumophila* sérotyp *Paris* jako na obrázku 42.

6.2.2 *Coxiella burnetii RSA 331* vs. *Beijerinckia indica ATCC 9039*

Další srovnávanou dvojicí je *Coxiella burnetii RSA 331* a *Beijerinckia indica ATCC 9039*. Sekvence *Coxiella burnetii RSA 331* kóduje 45 úseků, sekvence *Beijerinckia indica ATCC 9039* 39 úseků.

Zvolené parametry v mém programu byly následující: při prvním zarovnávání matice PAM250 a při druhém zarovnávání matice BLOSUM62, otvírací mezera -10, rozšiřovací mezera -1. Srovnávaná je sekvence aminokyselin.

Nastavení v programu PSAT bylo stejné jako v kapitole 6.2.1.

Výsledky:

V mém programu s thresholdem 30% a maticí PAM250 nebyla nalezena žádná shoda. První shoda byla zaznamenána až se snížením thresholdu na pouhých 22%, což už v žádném případě nelze považovat za homologii. Při změně použité matice na BLOSUM62 stále nebyla při thresholdu 30% zaznamenána žádná shoda, první shoda byla nalezena až se zvýšením thresholdu na 28%.

Programem PSAT bylo nalezeno 12 homologií.

6.2.3 *Francisella tularensis FSC198* vs. *Clostridium botulinum B1 Okra*

Poslední srovnávanou dvojicí *Francisella tularensis FSC198,* jejíž sekvence kóduje 1605 genových úseků a *Clostridium botulinum B1 Okra*, jejíž sekvence kóduje 194 úseků.

Zvolené parametry v mém programu byly následující: při prvním zarovnávání matice PAM250 a při druhém zarovnávání matice BLOSUM62, otvírací mezera -10, rozšiřovací mezera -1. Srovnávaná je sekvence aminokyselin.

Nastavení v programu PSAT bylo stejné jako v kapitole 6.2.1.

Výsledky:

Mnou vytvořený program při použití matice PAM250 nalezl 5 homologií s thresholdem 30% a při použití matice BLOSUM62 11 homologií (při tomtéž thresholdu).

Programem PSAT bylo nalezeno celkem 55 homologních sekvencí.

Celkové shrnutí: Při srovnání mnou vytvořeného programu a programu PSAT bylo zjištěno, že citlivost a přesnost nacházení homologií mým programem je menší, než programu PSAT. Celkový počet homologií byl vždy menší než u homologií nalezených PSAT. Nalezené homologie mým programem a programem PSAT pozičně odpovídaly, avšak procento nalezené identity se mírně lišilo (míra identity mnou vytvořeným programem byla přibližně o 10% nižší než programem PSAT). Ve srovnání matic, použitých pro srovnání, lepší výsledky vykazovala matice BLOSUM62, než matice PAM250.

Důvodem vyšší „úspěšnosti“ nacházení homologií programem PSAT je nejspíše zcela jiný algoritmus (PSAT nejspíše pracuje na principu lokální zarovnání), a proto nelze považovat nízký počet homologií nalezených mým programem za neúspěch, jelikož srovnání není zcela relevantní.

Závěr

Cílem této diplomové práce bylo pojednat o problematice homologií a algoritmech sloužících k jejich vyhledávání. Praktickým výstupem práce je program založený na globálním zarovnávání, tedy na algoritmu Needleman – Wunsch. K programu je také vytvořené uživatelské rozhraní. Algoritmus byl realizován v programovém prostředí Matlab.

Vytvořený program je schopen srovnávat sekvence jak nukleotidové, tak sekvence aminokyselin. Je možno v něm měnit vstupní podmínky zarovnání – substituční matici a hodnotu penalizace mezer. Také je možné navrhnout matici svoji vlastní. Dále umožňuje srovnat mezi sebou celé genomy či pouze určité sekvenční úseky. Limitací programu je však délka srovnávaných sekvencí, jelikož program je aktuálně vytvořen tak, že při výpočtu je zapojeno pouze jedno jádro procesoru počítače. Tato limitace tedy otvírá cestu pro vylepšení vzniklého programu – implementaci programu za použití Parallel Computing Toolboxu.

Správnost práce programu byla ověřena srovnáním s komerčně dostupnými programy pro globální zarovnávání nukleotidů – EMBOSS Needle a pro globální zarovnávání proteinů – nástroj ggsearch programu FASTA Sequence Comparison. Výsledky získané těmito komerčními programy se od mnou vytvořeného programu lišily pouze v řádu desetin, z čehož lze vyvozovat závěr, že program pracuje správně – parametrem tedy byla pouze správnost práce programu a provedení správného programu. Srovnávání např. časové či výpočetní náročnosti vzhledem k robustnosti kódu a pokročilosti kódu, v němž jsou komerční programy vytvořené, postrádalo význam.

Jako testovací sekvence byly použity volně dostupné sekvence z databáze NCBI. Program je optimalizován pouze pro srovnávání prokaryontních sekvencí, a to z důvodu nepřítomnosti intronů v sekvenci DNA – v případě eukaryotních organismů by bylo nezbytné ošetřit vystřihávání právě těchto nekódujících intronů.

Při samotném testování nastavení vstupních parametrů bylo prokázáno, že skutečně nejvhodnější maticí pro srovnávání nukleotidových sekvencí je matice *nuc44*, naopak nevhodnou se ukázala být matice identity a matice tranzitně transverzní.

Testování vlivu hodnot penalizací za vnesení přineslo následující výsledky: nejlepší zarovnání poskytuje matice *nuc44* s penalizací -10 pro otvírací mezeru a -0,5 pro rozšiřovací mezeru. Při zvolení jiných hodnot penalizace mezer již program nenacházel optimální zarovnání (konfrontováno s komerčními programy), ale výsledky byly zkreslené – buďto jsou některé mezery přeskakovány zcela a pak jsou nalezeny shody v místech, kde se reálně nenachází nebo jsou naopak mezery vkládány nadbytečně, čímž ve výsledku dochází k nenalezení některých shodných sekvencí.

V případě srovnávání sekvencí aminokyselin se ukázala být nejvhodnější matice BLOSUM62, opět s penalizací mezer -10 pro otvírací mezeru a -0,5 pro rozšiřovací mezeru.

Při testování programu bylo také potvrzeno, že je skutečně lepší vyhledávat homologie v sekvencích aminokyselin než v sekvencích nukleotidů. Výsledky nalezených shod mezi

sekvencemi nukleotidů mohou být zkreslené vzhledem k nízkému počtu možných kombinací bází – nalezená shoda tedy může být dílem náhody s mnohem větší pravděpodobností, než shoda nalezená v sekvenci aminokyselin.

Při konfrontaci mnou vytvořeného programu s programem PSAT vyhledávajícím homologie bylo prokázáno, že můj program nachází výrazně menší počet homologií než program PSAT. Toto srovnání však bohužel není zcela validní, jelikož PSAT pracuje na zcela jiném algoritmu než mnou vytvořený program, a tedy jsou jiné i výsledky, které poskytuje.

Srovnávání dvou sekvencí, jak nukleotidů, tak aminokyselin, je v mnoha případech pouhou „vstupní branou“ pro další práci mnoha nejen biologů a informatiků. V současné době, kdy do bioinformatických databází přibývá neustále větší množství nově osekvenovaných genomů a informací o nich, se dá tedy očekávat, že se i oblast zabývající se právě srovnáváním biologických sekvencí bude dále rozvíjet a nabývat na významu.

Seznam použitých zdrojů

[1] Flegr, J. *Evoluční biologie*. 2., opr. a rozš. vyd. Praha: Academia, 2009, 569 s. ISBN 978-80-200-1767-3.
[2] Gerlt, J.A. and P.C. Babbitt, *Can sequence determine function?* Genome Biol, 2000. 1(5): p. REVIEWS0005.
[3] Eisen, J.A., *Phylogenomics: improving functional predictions for uncharacterized genes by*. - Genome Res. 1998 Mar;8(3):163-7., (- 1088-9051 (Print)): p. T - ppublish.
[4] Krane, D. E. a Raymer M. L. *Fundamental concepts of bioinformatics*. San Francisco: Benjamin Cummings, 2003, xiii, 314 s. ISBN 08-053-4633-3.
[5] Larkin, M.A., et al., *Clustal W and Clustal X version 2.0.* - Bioinformatics. 2007 Nov 1;23(21):2947-8. Epub 2007 Sep 10., (- 1367-4811 (Electronic)): p. T - ppublish.
[6] Higgins, D.G. and P.M. Sharp, *CLUSTAL: a package for performing multiple sequence alignment on a microcomputer.* Gene, 1988. 73(1): p. 237-244.
[7] Ulbrichová I. *Nauka o lesním prostředí.* [online] 2010 [cit. 2014-19-5]. Dostupné z: http://fld.czu.cz/vyzkum/nauka_o_lp/chemie/chemie.html
[8] Lodish H, Berk A, Zipursky SL, et al. *Molecular Cell Biology*. 4th edition. New York: W. H. Freeman; 2000. Section 4.1, Structure of Nucleic Acids [online]. 2000 [cit. 2013-28-12]. Dostupné z: http://www.ncbi.nlm.nih.gov/books/NBK21514/
[9] Lodish, H., et al. *Molecular Cell Biology*. New York : W. H. Freedman and Company, 2004.
[10] Burks, C., et al., *[1] GenBank: Current status and future directions*, in *Methods Enzymol* 1990, Academic Press. p. 3-22.
[11] Šmarda, J. *Metody molekulární biologie*. 1. vyd., 2. dotisk. Brno: Masarykova univerzita, 2010 , 188 s. ISBN 978-80-210-3841-7.
[12] Stoesser, G., et al. *The EMBL Nucleotide Sequence Database.* Nucleic Acids Res. 2002 Jan 1;30(1):21-6.
[13] Cvrčková, F. *Úvod do praktické bioinformatiky.* 1. vyd. Praha: Academia, 2006. 148 stran. ISBN 80-200-1360-1.
[14] Borodovsky, M. and Ekisheva, S. *Problems and Solutions in Biological Sequence Analysis.* 1. vyd. New York: Cambridge University Press, 2006. 346 stran. ISBN-13 978-0-521-84754-4
[15] Sarıyer, O.S. and C. Güven, *Sequence alignment using simulated annealing.* Physica A: Statistical Mechanics and its Applications, 2010. 389(15): p. 3007-3012.
[16] Murata, M., *[22] Three-way Needleman—Wunsch algorithm*, in *Methods Enzymol*1990, Academic Press. p. 365-375.
[17] Smith T. F. a Waterman M. S. *Identification of Common Molecular Subsequences*. Journal of Molecular Biology. 1981, vol. 147, s. 195–197.
[18] Ma, B., J. Tromp, and M. Li, *PatternHunter: faster and more sensitive homology search.* Bioinformatics, 2002. 18(3): p. 440-5.
[19] Xiong, J. *Essential Bioinformatics*. Cambridge University Press, 2006. ISBN 978-0-521-60082-8.
[20] Altschul, S.F., et al., *Basic local alignment search tool.* J Mol Biol, 1990. 215(3): p. 403-410.
[21] Baxevanis, A. D. a Ouellette B. *Bioinformatics: a practical guide to the analysis of genes and proteins*. 2nd ed. New York: Wiley-Interscience, c2001, xviii, 470 p., [13] p. of plates. ISBN 04-713-8391-0.
[22] Casey, R.M. BLAST sequences aid in genomics and proteomics. *Business Intelligence Network* [online]. 2005 [cit. 2014-11-04]. Dostupné z: http://www.b-eye-network.com/view/1730.

[23] Mount, D. W. *Bioinformatics: Sequence and Genome Analysis* (druhé vydání). Cold Spring Harbor Press. 2006. ISBN 978-0-8796-9712-9.
[24] Brudno, M., et al. *LAGAN and Multi-LAGAN: Efficient Tools for Large-Scale Multiple Alignment of Genomic DNA*. Genome Research. 2002, vol. 13, issue 4, p. 721-731.
[25] Bray, N., Dubchak I. and Pachter L. AVID: *A Global Alignment Program*. Genome Research. vol. 13, issue 1, s. 97-102.
[26] Jonassen, I. and Junhyong K. *Algorithms in bioinformatics: 4th international workshop, WABI 2004, Bergen, Norway, September 17-21, 2004 : proceedings*. New York: Springer, c2004, ix, 476 p. Lecture notes in computer science. ISBN 35-402-3018-1.
[27] Snustad, D. P. a Simmons M. J.. *Genetika*. Vyd. 1. Překlad Jiřina Relichová. Brno: Masarykova univerzita, 2009, xxi, 871 s. ISBN 978-802-1048-522.
[28] Kulikova, T., Aldebert, P., Althrope, N., Baker, W., Bates, K., et al. *The EMBL Nucleotide Sequence Database.* Nucleic Acid Res. 2004. 32, D27-D30.
[29] Bairoch, A. *Swiss-Prot: Juggling between evolution and stability.* Briefings in Bioinformatics. 2004-01-01, vol. 5, issue 1, s. 39-55
[30] Kouranov, A., et al. *The RCSB PDB information portal for structural genomics.* Nucleic Acids Research. 2006-01-01, vol. 34, issue 90001, D302-D305.
[31] Henikoff, S. *Scores for sequence searches and alignments.* Current Opinion in Structural Biology. 1996, vol. 6, issue 3, s. 353-360.
[32] Vingron, M. a Waterman, M.S. *Sequence alignment and penalty choice.* Journal of Molecular Biology. 1994, vol. 235, issue 1, s. 1-12.
[33] Campanella, J.J., Bitincka L. a Smalley J. *Sequence alignment and penalty choice*. BMC Bioinformatics. 1994, vol. 4, issue 1, s. 29-.
[34] Dayhoff, M. O., Barker W. C. a Hunt L. T. *Establishing homologies in protein sequences*. Methods in enzymology, vol. 91, s. 524.
[35] Pearson, W. R., Barker W. C. a Hunt L. T. *Selecting the Right Similarity-Scoring Matrix*. Current Protocols in Bioinformatics. Hoboken, NJ, USA: John Wiley, 2002-08-15, 3.5.1.
[36] Dayhoff M. O., Schwartz R. M. a Orcutt B. C. *A model of evolutionary change in proteins.* Atlas of Protein Sequence and Structure, vol 5, suppl 3. National Biomedical Research Foundation; 1978:345-358.
[37] Gonnet G. H., Cohen M. A. a Benner S. A. *Exhaustive matching of the entire protein sequence database.* Science, 1992, 256:1443-1445.
[38] Henikoff, S.; Henikoff, J.G. (1992). *Amino Acid Substitution Matrices from Protein Blocks*. PNAS 89 (22): 10915–10919.
[39] Dayhoff, M. G. a Eck, R. V. *Atlas of Protein Sequence and Structure.* Natd. Biomed. Res. Found., Silver Spring, MD, 1968, Vol. 3, p. 33.
[40] StyczynskiI, M. P., et al. *BLOSUM62 miscalculations improve search performance.* Nature Biotechnology. 2008, vol. 26, issue 3, s. 274-275.
[41] Johns Hopkins University. Homework 1b, Due Nov 7, 2005 [online]. 2013 [cit. 2014-05-11]. Dostupné z: http://astor.som.jhmi.edu/~sining/BSA/hw/hw01b.htm
[42] Needleman, S. B. a Wunsch Ch. D. *A general method applicable to the search for similarities in the amino acid sequence of two proteins.* Journal of Molecular Biology. 1970, vol. 48, issue 3, s. 443-453.
[43] Zvelebil, M. a Baum J. O. *Understanding bioinformatics*. New York: Garland Science, 2008, xxiii, 772 s. ISBN 978-0-8153-4024-9.
[44] Complete Software for Proteomics. Nov 30, 2013 [online]. 2013 [cit. 2014-05-17]. Dostupné z: http://www.bioinfor.com/images/stories/pdf/patternhunterbrochure.pdf
[45] Thompson, J. D., Higginds D. G. A Gibson T. J. *CLUSTAL W: improving the sensitivity of progressive multiple sequence alignment through sequence weighting, position-specific gap*

penalties and weight matrix choice. Nucleic Acids Research. 1994, vol. 22, issue 22, s. 4673-4680.
[46] Barton, G. J., Higgins D. G. a T. J. *Creation and Analysis of Protein Multiple Sequence Alignments: improving the sensitivity of progressive multiple sequence alignment through sequence weighting, position-specific gap penalties and weight matrix choice*. Bioinformatics. New York, USA: John Wiley, 2001-04-20, vol. 64, issue 1, s. 215.
[47] Carroll, S. B. a With illustrations by Carroll J. W. *Endless forms most beautiful: the new science of evo devo and the making of the animal kingdom*. 1st ed. published in New York, by W.W. Norton. London: Weidenfeld, 2006. ISBN 02-978-5094-6.
[48] Woese, C. R., Kandler O. a Wheelis M. L. *Towards a natural system of organisms: proposal for the domains Archaea, Bacteria, and Eucarya.* Proceedings of the National Academy of Sciences. 1990-06-15, vol. 87, issue 12, s. 4576-4579.
[49] Wolfe, K. *Robustness—it's not where you think it is.* Nature Genetics. vol. 25, issue 1, s. 3-4.
[50] Fitch, W. M. *Distinguishing Homologous from Analogous Proteins*. Systematic Zoology. 1970, vol. 19, issue 2, s. 99-.
[51] Xia, X. *Comparative genomics.* New York: Springer, 2013, p. cm. ISBN ISBN 978-3-642-37145-5.
[52] Karlin, S. a Altschul, S.F. *Methods for assessing the statistical significance of molecular sequence features by using general scoring schemes*. Proc. Natl. Acad. Sci. USA. 1990, vol. 87, s. 2264-2268.
[53] Altschul, S.F. *Amino acid substitution matrices from an information theoretic perspective*. J. Mol. Biol. 1991, vol. 219, s. 555-565.

Seznam zkratek

A	Adenin
BLAST	Basic Local Alignment Search Tool – Základní nástroj pro lokální zarovnání
BLOSUM	Blocks of Amino Acid Substitution Matrix – Matice substituovaných bloků aminokyselin
C	Cytosin
CIB	Center for Information Biology – Centrum pro informační biologio
DDBJ	DNA Data Bank of Japan
DNA	Deoxyribonucleic Acid – Deoxyribonukleová kyselina
EBI	Energy Biosciences Institute – Institut biologických věd
EMBL	European Molecular Biology Laboratory – Evropská molekulárně biologická laboratoř
EVD	Extreme Value Distribution – Extrémní rozdělení hodnot
FASTA	„FAST-All“ – „Vše rychle“
G	Guanin
GI number	GenBank Identifier – identifikátor v GenBank
HUGO	Human Genom Project – Projekt lidského genomu
IUPAC	The International Union of Pure and Applied Chemistry – Mezinárodní unie pro čistou a užitou chemii
LAGAN	Limited Area Global Alignment of Nucleotides – Omezené prostorové globální zarovnání nukleotidů
M-LAGAN	Multiple Limited Area Global Alignment of Nucleotides – Mnohonásobné omezené prostorové globální zarovnání nukleotidů
NCBI	National Center for Biotechnology Information – Národní centrum pro biotechnologické informace
NIG	National Genetic Institute – Národní genetický institut
NJ	Neighbour Joining – Sousední propojování
NMR	Nucler Magnetic Resonance – Nukleární magnetická rezonance
PAM	Point Accepted Mutation – Přípustná bodová mutace
PDB	Protein Data Bank – Proteinová databanka
PSAT	Prokaryotic Sequence Analysis Tool – Nástroj na analýzu sekvencí prokaryot
PSI BLAST	Position-Specific Iterative BLAST – Pozičně specifický interativní BLAST
RNA	Ribonucleic Acid – ribonukleová kyselina
RCSB PDB	Research Collaboratoryfor Structural Bioinformatics Protein Data Bank – Výzkumná spolupráce pro strukturní bioinformatiku – databanka proteinů
S-LAGAN	Shuffle Limited Area Global Alignment of Nucleotides – „Obejité“ omezené prostorové globální zarovnání nukleotidů
T	Thymin
TREMBL	Translated European Molecular Biology Laboratory – „Přeložená“ evropská Molekulárně biologická laboratoř
UPGMA	Unweighted Pair Group Method using Arithmetic – Shluková analýza
UNIPROT	Universal Protein Resource Knowleadgebase – Celosvětový zdroj proteinů

Seznam obrázků

Seznam tabulek