VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÝCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER SYSTEMS

VYHLEDÁVÁNÍ TRIPLEXŮ V DNA SEKVENCÍCH

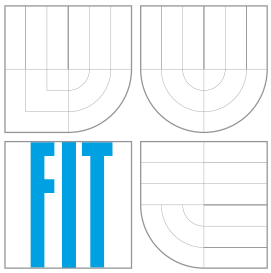
DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE
AUTHOR

Bc. MICHAL ZRŮNA

BRNO 2012

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÝCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER SYSTEMS

VYHLEDÁVÁNÍ TRIPLEXŮ V DNA SEKVENCÍCH

TRIPLEX DETECTION IN DNA SEQUENCES

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE
AUTHOR

Bc. MICHAL ZRŮNA

VEDOUCÍ PRÁCE
SUPERVISOR

Ing. TOMÁŠ MARTÍNEK, Ph.D.

BRNO 2012

Abstrakt

Studium triplexů má pro biologii, biotechnologii a medicínu velký význam, protože triplexy ovlivňují nejdůležitější procesy odehrávající se v DNA, jakými jsou replikace, transkripce, rekombinace a mutace. Tato diplomová práce obsahuje teoretickou část shrnující současné znalosti o triplexech a přehled existujících přístupů k detekci triplexů v DNA sekvencích. Přístup založený na metodě dynamického programování je implementován, rozšířen a vylepšen o nové rysy tak, aby byl použitelný pro vytvořenou webovou aplikaci, která pro hledání triplexů nabízí grafické uživatelské rozhraní a možnost triplexy vizualizovat.

Abstract

Study of triplexes is of great importance for biology, biotechnology and medicine because triplexes in DNA influence the most important processes such as replication, transcription, recombination and mutation. This master thesis includes theoretical chapters, that recap current knowledge of triplexes and present state of the art of triplex detection approaches. An approach based on a dynamic programming metod is implemented, extended and improved. This program is used in created web-based triplex search tool, that offers graphical user interface and triplex visulizations.

Klíčová slova

DNA, prohledávání DNA sekvencí, triplexy, hledání triplexů, detekce triplexů, vizualizace triplexů

Keywords

DNA, searching DNA sequences, triplexes, searching for triplex structures, triplex detection, triplex visualization

Citace

Michal Zrůna: Vyhledávání triplexů v DNA sekvencích, diplomová práce, Brno, FIT VUT v Brně, 2012

Vyhledávání triplexů v DNA sekvencích

Prohlášení

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně pod vedením pana Ing. Tomáše Martínka, PhD.

.....
Michal Zrůna
22. května 2012

Poděkování

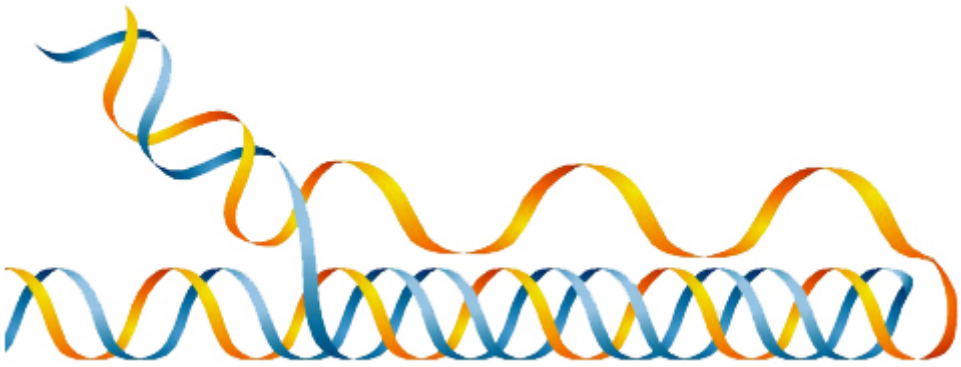
Touto cestou děkuji svému vedoucímu za vstřícný přístup, ochotu cokoliv vysvětlit a pomoc při shánění materiálů potřebných k vypracovýní této práce.

Obsah

Kapitola 1

Úvod

Výzkum nad nestandardními strukturami DNA je veden řadu let. Jednou z možných nestandardních forem DNA je i třívláknová struktura označovaná jako triplex (ukázka triplexu na obrázku 1.1). O triplexech bylo doposud zjištěno, že ovlivňují nejdůležitější procesy odehrávající se v DNA, jakými jsou replikace, transkripce, rekombinace a mutace. Jak přesně tento mechanismus funguje, však doposud zjištěno nebylo.[14] Nicméně i přesto, že přesný princip těchto mechanismů neznáme, umíme jich do jisté míry už dnes využívat. Jednou z vyvíjených metod jsou tzv. *TFO (triplex-forming oligonucleotides)*. Biologové jsou schopni uměle syntetizovat krátké sekvence nukleotidů, které se na specifických místech sekvence DNA naváží a vynutí formaci triplexu, čímž dokážou například potlačit transkripci. Tato metoda má zatím z chemického hlediska své nedostatky, ale předpokládá se, že v budoucnu by mohla být úspěšně využívána v genové terapii. Kromě problémů chemického charakteru je výrazný problém i detekce míst v DNA, na které by se *TFO* mohly vázat[14]. S tímto problémem se lze úspěšně vypořádat prováděním analýz DNA sekvencí na počítačích. I přesto, že dnes neexistuje žádný model, který by bral v úvahu veškeré okolnosti ovlivňující vznik triplexu, dokážeme pomocí zjednodušeného modelu najít místa, kde by alespoň s velkou pravděpodobností vzniknout mohl.



Obrázek 1.1: *Ukázka intramolekulárního triplexu převzatá z [14].*

Cílem této diplomové práce je prostudovat základní principy molekulární biologie, blíže se zaměřit na třívláknové struktury, prozkoumat existující algoritmy pro vyhledávání takových struktur a jeden z těchto algoritmů implementovat.

Každý přístup k vyhledávání potenciálních triplexů v DNA sekvencích je specializovaný

na jiný typ triplexů. Jejich základní myšlenka je ve své podstatě podobná. Obvykle se program během analýzy snaží hledat homopurinové/homopyrimidinové sekvence. Odlišnosti vznikají až ve způsobu jakým je hledají a v tom, jak přísné podmínky stanovují. Většina přístupů je striktně zaměřena na vyhledávání téměř dokonalých úseků pro tvorbu triplexu, avšak podle literatury existují i ne tak dokonalé a přesto stabilní triplexy. Detekci takových triplexů bere v úvahu až (Lexa, 2011)[16]. Blíže o jednotlivých přístupech a analýzách pojednává kapitola 3.

Jádro vybraného způsobu vyhledávání triplexů tvoří algoritmus převzatý z (Lexa, 2011)[16]. Tento algoritmus vychází z algoritmu pro hledání palindromů a je založen na principech dynamického programování. Tato práce ho v některých aspektech rozšiřuje. Asi nejdůležitější rozšíření spočívá v nově zavedené schopnosti programu zpracovat veškeré povolené symboly z FASTA formátu a dosadit za ně nejvhodnější nukleotid. Detailněji veškeré úpravy rozebírá kapitola 4. Kromě rozšíření původního algoritmu byla v rámci této práce vytvořena i modifikovaná verze celého programu. Ten v původní verzi nepracuje s celou maticí dynamického programování, protože celá matice by zabírala velké množství paměti. Z toho pak plyne problém, jak udělat zpětný průchod touto maticí, který je potřebný pro získání přesné sekvence triplexu včetně inzercí. Modifikovaná verze programu je určena právě pro tento problém, během výpočtu je postupně matice sestavena, takže následně je snadné udělat zpětný průchod a přesnou sekvenci zjistit.

Na výše zmíněném programu je postavena i webová aplikace pro hledání triplexů v zadaných sekvencích vytvořená také v rámci této práce. Kromě klasického výčtu výsledků aplikace rovnou vytváří i schematické vizualizace nalezených triplexů, nabízí jejich zobrazení v UCSC genome browseru nebo stažení v gff3 formátu. Navíc je v aplikaci možnost použít nástroj pro vizualizaci triplexů i samostatně a vizualizovat si tak vlastní triplex.

Podrobnější teorie ohledně obecné struktury DNA, triplexů a jejich využití je rozepsána v kapitole 2. Kapitola 3 pojednává blíže o jednotlivých přístupech a analýzách zaměřených na hledání triplexů. Kapitoly 4 a 5 popisují konzolovou a webovou aplikaci v rámci této práce vytvořené. V předposlední kapitole jsou představeny provedené experimenty. Poslední kapitola zhodnocuje dosažené výsledky a navrhuje další vylepšení, která by ve vytvořených aplikacích v budoucnu mohla být zavedena.

Kapitola 2

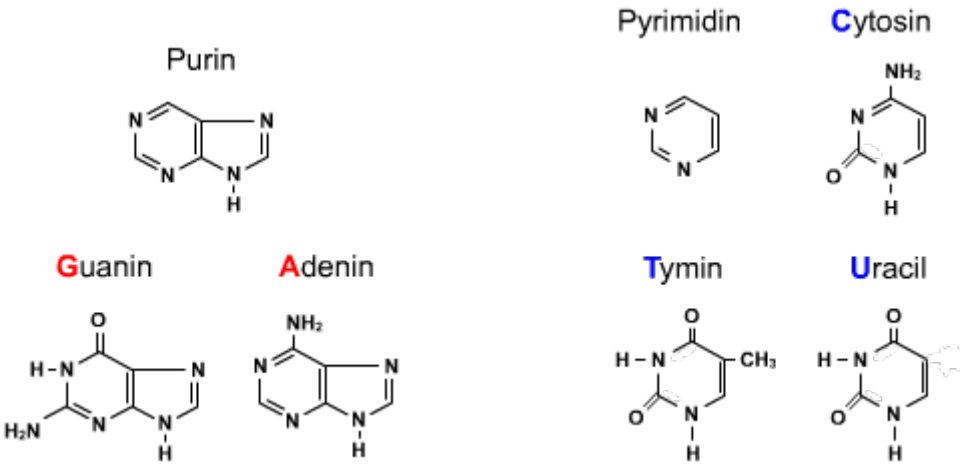
Struktura DNA a triplexy

Deoxyribonukleová kyselina (DNA) je nosičem genetické informace. Během života organismu je genetická informace nespočetněkrát kopírována a přenášena z buňky do buňky velmi důmyslným a precizním mechanismem, který byl pro vědeckou obec záhadou až do roku 1953. Teprve v tomto roce představili James Watson a Francis Crick model struktury DNA, ačkoliv o DNA samotné se vědělo už od roku 1868, kdy ji objevil Friedrich Miescher. Za tento převratný objev se jim také v roce 1962 dostalo uznání v podobě Nobelovy ceny. Odhalení struktury DNA vedlo k pochopení toho, jak se DNA replikuje a jak by mohla kódovat instrukce pro tvorbu proteinů.

2.1 Struktura DNA

Molekula DNA je složena ze dvou dlouhých polynukleotidových vláken svinutých do dvojšroubovice. Každý nukleotid je tvořen zbytkem kyseliny fosforečné a sacharidem deoxyribózou, ke kterému je připojena jedna ze čtyř nukleotidových bází — adenin (A), guanin (G), cytosin (C), tymin (T) (viz obrázek 2.1). Nukleotidy v jednotlivých vláknech jsou spojeny kovalentní vazbou mezi sacharidem a fosfátem, zatímco dvojšroubovici udržují vodíkové můstky mezi protilehlými nukleotidovými bázemi, báze tedy směřují dovnitř dvojšroubovice. Párování bází není náhodné, za běžných podmínek se řídí pravidly, která poprvé představili pánové Watson a Crick. Podle nich se tomuto způsobu párování říká Watsonovo-Crickovo párování bází. Nicméně existují i alternativní způsoby párování, které však vyžadují zvláštní podmínky, anebo se uplatňují u RNA, zatímco u DNA ne. Jedním z takových alternativních druhů párování je i Hoogstenovo a reverzní Hoogstenovo párování, jež hraje velkou roli při tvorbě triplexů (viz dále). Díky tomu, že známe způsob, jakým se báze párují, můžeme ze sekvence jednoho vlákna přesně zjistit sekvenci druhého vlákna. Této vlastnosti říkáme **komplementarita vláken**. Komplementarita má zásadní význam při kopírování DNA.

Každé vlákno DNA má svou **orientaci**, již určujeme podle toho, zdali končí OH skupinou na sacharidu (tento konec vlákna nazýváme $3'$ konec) nebo fosfátovou skupinou ($5'$ konec). V dvojšroubovici DNA mají vlákna navzájem opačný směr, říkáme, že jsou antiparalelní a označujeme je jako $+$ a $-$. Kromě směru vláken je pro dvojšroubovici taktéž důležité, jaké má vinutí. Většinou se setkáváme s pravotočivým vinutím (A-DNA, B-DNA), méně častou variantou je opačné levotočivé vinutí (Z-DNA). Ukázka dvojšroubovice DNA včetně znázornění zde zmíněných pojmů je na obrázku 2.3, který je převzatý z [1]. Ačkoliv tento obrázek zastupuje nejběžnější formu DNA, obecně se molekula DNA v daném pro-



(a) puriny (b) pyrimidiny

Obrázek 2.1: *Nukleotidové báze dělíme na dvě skupiny podle toho, jestli jejich základ tvoří pyrimidin nebo purin.*

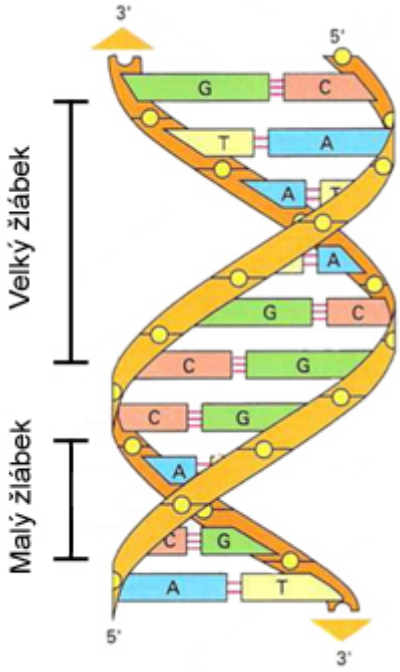
(a) A-DNA (b) B-DNA (c) Z-DNA

Obrázek 2.2: *Možné konformace molekuly DNA převzaté z [20]*

středí snaží vždy zaujmout energeticky nejvýhodnější pozici, což znamená, že v různých prostředích můžeme narazit na mírně odlišné konformace molekuly DNA [13]:

- `A-DNA` pravotočivá, krátká a široká molekula 2.2a
- `B-DNA` pravotočivá, delší ale tenčí molekula 2.2b
- `Z-DNA` levotočivá, dlouhá, tenká molekula 2.2c

Kromě **primární struktury** (sekvence nukleotidů) a **sekundární struktury** (dvojšroubovice) hovoříme u DNA i o **terciální struktuře**. Terciální struktura označovaná jako nadšroubovice neboli superhelix vzniká přidáním dalšího vinutí do dvojšroubovice nebo jejím svinutím okolo určitých proteinů (viz dále).



Obrázek 2.3: **Pravotočivá dvojřetězcová molekula DNA** - *pevné kovalentní vazby mezi zbytky kyseliny fosforečné a sacharidy v kostře dvojšroubovice jsou symbolicky znázorněny tím, jak do sebe tyto dvě části zapadají (žluté kolečka zapadají do jamek). Červené čárky mezi nukleotidovými bázemi znázorňují vodíkové můstky. Mezi dvojicemi tvořenými C a G jsou tři vodíkové můstky, zatímco u párů tvořených z T a A jsou pouze dva.*

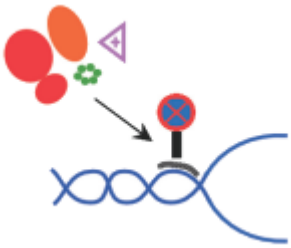
2.2 Triplexy

Dvojšroubovice není jediná možná struktura, kterou DNA tvoří. Bylo zjištěno, že DNA je schopna vytvářet i jiné, ne tak běžné struktury. Z biochemického hlediska jsou takové struktury stabilní za přítomnosti multivalentních kationtů a v oblastech s vysokou mírou záporného nadšroubovicového vinutí. Z pohledu genetického se obvykle jedná o místa s nějakým specifickým motivem jako např. vysokým počtem repetic.[19] Většina z těchto alternativních struktur je tvořena jen přechodně při určitých genetických procesech často v místech, která mají dopad na funkci genu. Mezi takovéto struktury patří např. křížová struktura, vlásenková struktura (hairpin), G-tetráda (kvadruplex), trvale rozvinutá DNA a pro tuto práci zásadní třívláknová struktura tzv. **triplex**.

V roce 1957 ukázali Felsenfeld, Davies a Rich jak za vhodných podmínek může vzniknout třívláknová molekula ribonukleové kyseliny. Tento RNA triplex byl vytvořen z poly(A)·poly(U) dvojšroubovice a třetího poly(U) vlákna.[18] Velký žlábek DNA obsahuje skupiny donorů a akceptorů, schopných vytvořit vodíkové můstky s třetím vláknem. Interakce vodíkových můstků jsou v těchto místech odlišné od Watsonových-Crickových, které udržují dvojšroubovici. Označujeme je jako Hoogsteenovy vodíkové můstky. Plný potenciál tohoto objevu byl však pochopen až o třicet let později, když bylo zjištěno, že krátké oligonukleotidy mohou díky Hoogsteenovým můstkům ve velkém žlábku vytvořit vazbu k dvojšroubovici DNA, a tak zformovat trojšroubovicovou strukturu. Takové oligonukleotidy označujeme jako *triplex-forming oligonucleotides*, tedy oligonukleotidy tvořící triplexy (dále jen TFO).

(a) Vzniklý triplex fyzicky blokuje transkripci a prodlužování replikační vidličky.

(b) Triplex vynutí vazbu nějakého DNA modifikujícího agenta např. restrikční endonukleázy.



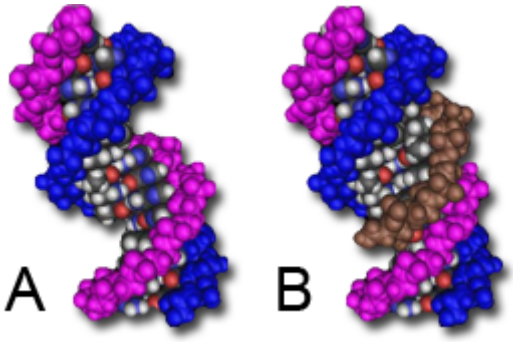
(c) Triplex brání zahájení transkripce případně replikace.

Obrázek 2.4: *Příklady užití triplexů k regulaci genové exprese. Převzato z [7]*

Zároveň vědecké týmy okolo Petra Dervana, Claude Helene, Jacqua Fresca a Roberta Wellse ukázaly, že pomocí takových oligonukleotidů lze na specifických místech vynutit rozštěpení dvojšroubovice a po bližším studiu poprvé navrhly možnost, že by triplexy mohly hrát roli v regulaci genové exprese. TFO dnes představují zajímavou a používanou možnost, jak cíleně zaměřit určité úseky v DNA. Mezi příklady užití patří vázání transkripčního faktoru na specifický úsek DNA nebo řízená mutageneze a modifikace DNA s cílem regulovat genovou expresi (obrázek 2.4).[7]

2.3 Vznik a rozdělení triplexů

Vznik triplexu je úzce spjat s úseky DNA, které se vyznačují velkým zastoupením nukleotidů s buďto purinovou nebo pyrimidinovou bází. Třetí vlákno se k dvojšroubovici váže v oblasti velkého žlábku tak, jak je ilustrováno na obrázku 2.5 a jeho nukleotidy jsou k dvojšroubovici vázány podle pravidel Hoogsteenova párování bází (obrázek 2.6).[5] Zatímco v laboratorních podmínkách je vytvoření triplexu celkem přímočaré, v jádře živé buňky existuje mnoho komplikací, které musí třetí vlákno překonat. Musí odolat nukleázám, překonat odpudivé síly vyvolané záporným nábojem mezi dvojšroubovicí a sebou samotným, být schopno vytvořit triplex v pH přirozeném pro prostředí buňky aj. Z to-



Obrázek 2.5: *A — Prostorový model dvojšroubovice DNA. B — Prostorový model dvojšroubovice s navázaným třetím vláknem ve velkém žlábku tvořící tak triplex. Obrázek převzat z [5].*

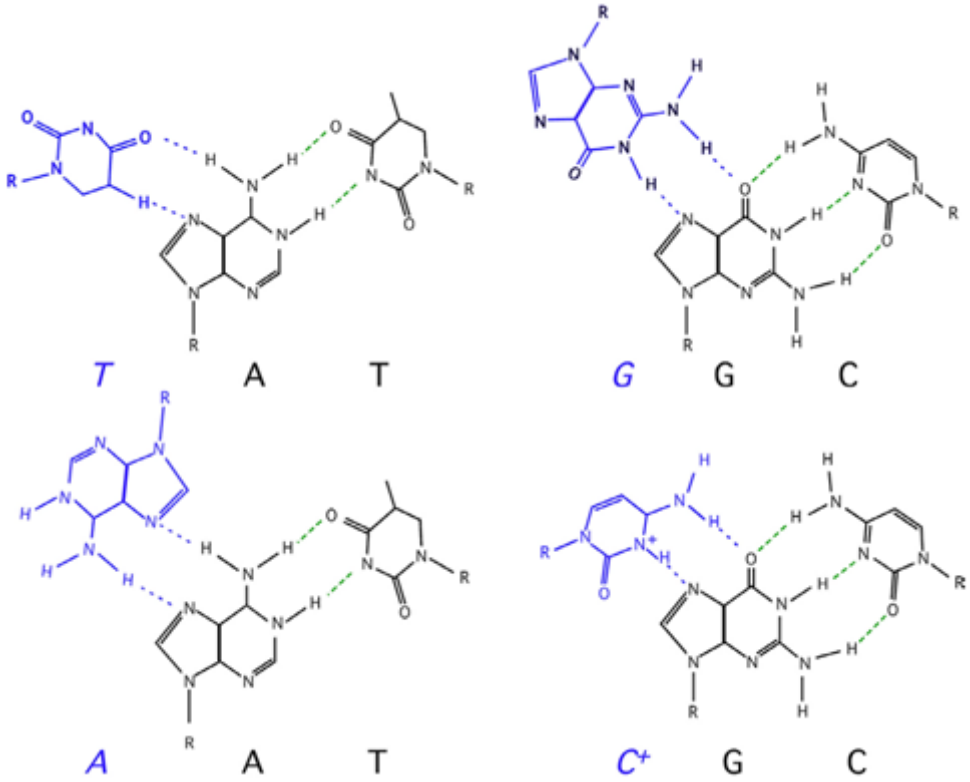
hoto důvodu je nutné při experimentech vynucujících tvorbu triplexu báze a kostru TFO chemicky modifikovat.[7]

Triplexy rozdělujeme podle původu a složení třetího vlákna. V pyrimidinovém motivu (Y) je třetí vlákno složené z pyrimidinových bází paralelně vázané k purinovému vláknu dvojšroubovice pomocí Hoogsteenových vodíkových můstků a je stabilní při mírně kyselém pH, které napomáhá protonaci cytosinu. V tomto motivu převládají izomorfní T·A·T a C^+·G·C triplety. V purinovém motivu (R) je třetí vlákno homopurinové a k purinovému vláknu dvojšroubovice je navázáno díky reverzním-Hoogsteenovým vodíkovým můstkům. V R motivu převládají T· (nebo A·)A·T a G·G·C triplety. Zmíněné trojice, které se účastní tvorby triplexů, jsou ilustrovány na obrázku 2.6. Na rozdíl od Y motivu je R motiv relativně nezávislý na pH.[12][16][5] Z hlediska původu třetího vlákna uvažujeme dělení na *intramolekulární* a *intermolekulární triplexy*. Třetí vlákno může pocházet ze stejné molekuly DNA, v takovém případě je triplex označován jako intramolekulární (též označováno jako H–DNA, případně *H–DNA podle orientace třetího vlákna), nebo z jiné molekuly DNA, pak jej nazýváme intermolekulární. Oba typy triplexů mohou vzniknout jak v Y tak v R motivu. U intramolekulárních triplexů lze dále rozlišit, ze kterého vlákna je třetí vlákno do triplexu dodáno a podle tohoto kritéria rozlišovat mezi *intrastrand* triplexy (celý triplex je tvořen svinutím pouze jednoho vlákna) a *interstrand* triplexy (třetí vlákno triplexu pochází z komplementárního vlákna).[12][16][5]

2.4 Využití triplexů

Intermolekulární triplexy tvořené TFO (obrázek 2.7) mají zejména díky své schopnosti zaměřit určité geny a měnit jejich strukturu nebo funkci přímo v genomu obrovský přínos pro biologii, biotechnologii a medicínu. TFO díky velké afinitě[1] a schopnosti vázat se na dvojšroubovici DNA na specifických místech představují téměř ideální molekuly pro tento účel. Navíc lze TFO snadno chemicky modifikovat, což například umožňuje k nim navázat činitele poškozující DNA a provést tak cílené poškození nějaké oblasti v genomu. Kromě oligonukleotidů určených k zaměření nukleových kyselin (TFO), existují i oligonukleotidy vytvořené tak, aby jejich cílem byly proteiny (PTO). V tomto směru přináší TFO potenciálně výhodu, protože modifikací/mutací genu, ovlivníme veškeré jeho nové produkty nebo

[1] Chemická afinita popisuje ochotu atomu nebo sloučeniny reagovat s jiným atomem nebo sloučeninou.[21]



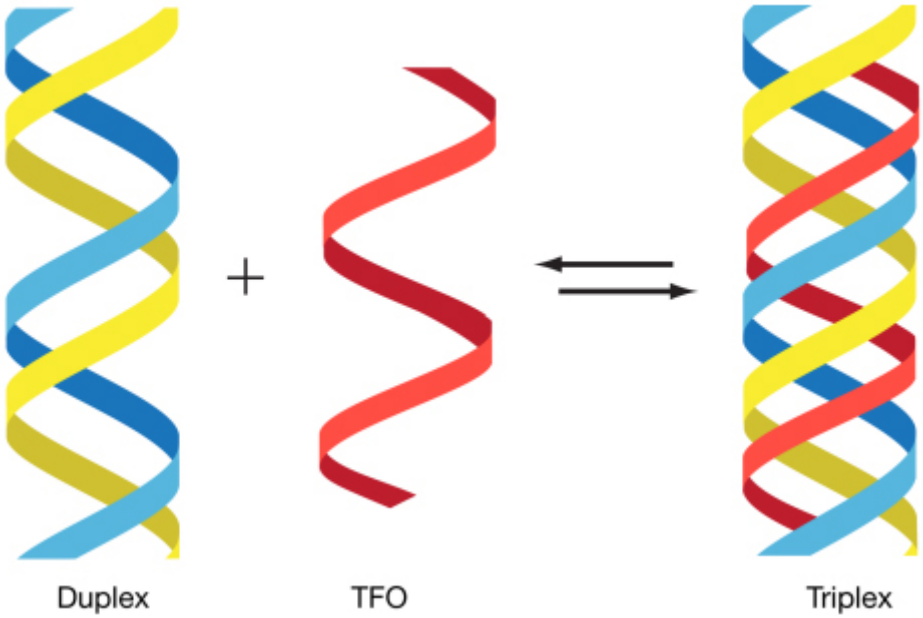
Obrázek 2.6: *Trojice nukleotidů objevující se v triplexech tzv. triplety. Watsonovo-Crickovo párování je zobrazeno zelenými čárkami, Hoogsteenovo párování je zobrazeno modře. Obrázek převzat z [5].*

ho rovnou můžeme deaktivovat. Zatímco použitím PTO ovlivníme pouze omezený počet jeho produktů.[14]

Aby bylo možné manipulovat se strukturou a nebo s funkcí genu, musí tento gen obsahovat oblast, na níž je TFO schopno se navázat. Tato místa lze určit analýzou genomu. Podrobnější popis takové analýzy je popsán dále v kapitole 3.1. V lidském i myším genomu je takových oblastí zejména v promotorech a transkribovaných oblastech velké množství. Potlačení transkripce pomocí TFO bylo mnohokrát demonstrováno (poprvé vědeckým týmem okolo M. E. Hogana), nicméně narazilo na jisté potíže. Tyto problémy se netýkají pouze pokusů o potlačení transkripce, nýbrž obecně celé TFO technologie. Konkrétně se jedná o problém jak dopravit TFO do buňky a jak zajistit jeho stabilitu, jakmile je uvnitř buňky. Dále může představovat problém nedostatečná afinita vazebného místa způsobená nevhodným pH a koncentrací solí v mezibuněčném prostoru nebo se může stát, že vazebné místo je zrovna nedostupné kvůli chromatinové bariéře. Také se může stát, že TFO místo vazby na námi požadované místo zaujme zcela jinou činnost. Existuje totiž možnost použít TFO jako falešný cíl pro vazbu transkripčních faktorů, které se tím pádem už nemohou vázat nikam na dvojšroubovici a zahájit transkripci. Tento jev paradoxně také vede k potlačení transkripce, ale s velkou pravděpodobností jiného genu než jsme požadovali za předpokladu, že naším cílem bylo zabránit transkripci. Kromě potlačení transkripce lze TFO použít pro záměrné poškození DNA, což jak bylo zjištěno, stimuluje mutaci, rekombinaci a opravné mechanismy DNA na poškozeném úseku.[14]

Pro zlepšení afinity, selektivity a stability oligonukleotidů uvnitř buňky lze provést různé modifikace bází, cukrfosfátové kostry nebo $5'/3'$ konce. Mezi používané modifikované báze patří například 6-thioguanin používaný namísto guaninu nebo 7-dezaxanthin nahrazující adenin. Obě dvě jmenované náhražky jsou používány v TFO navržených k tvorbě antiparalelního R triplexu a brání tvorbě nežádoucích sekundárních struktur uvnitř TFO. Při tvorbě paralelního Y triplexu bylo demonstrováno nahrazení cytosinu 5-metylcytosinem za účelem snížit závislost tvorby triplexu na pH. Jako poslední příklad modifikované báze zmiňme 5-propynyluracil, který slouží jako náhražka tyminu a zajišťuje větší stabilitu triplexu.

Konkrétní chemické modifikace cukrfosfátové kostry zde nebudou uvedeny, postačí zmínit jejich účel. Kostra TFO i dvojšroubovice DNA má záporný náboj, navzájem se tedy odpuzují a to ztěžuje tvorbu triplexu. Modifikace kostry způsobí změnu jejího náboje na neutrální nebo kladný, a tak usnadní tvorbu triplexu. Poslední druh úpravy TFO spočívá v modifikaci 5' nebo 3' konce. S pomocí takové modifikace lze dosáhnout vyšší odolnosti TFO proti degradaci způsobené exonukleázami.[14][10]



Obrázek 2.7: *V dvojšroubovici (duplexu) je purinové vlákno naznačeno modře a pyrimidinové žlutě. TFO, který se na duplex váže v oblasti velkého žlábku, je znázorněn červeně. Převzato z [14].*

Genomy eukaryotických buněk obsahují mnoho oblastí citlivých na nukleázu S1. Typickým rysem takových oblastí jsou polypurinové–polypyrimidinové úseky. Tyto úseky mají potenciál vytvořit **intramolekulární** triplex nazývaný též **H–DNA** (ukázka na obrázku 1.1 v úvodní kapitole). Nicméně přímá detekce H–DNA v genomech eukaryotických buněk je příliš složitá. Proto biologové k pozorování H–DNA in vivo používají bakterii E. coli s vpravenými plasmidy, které obsahují sekvence schopné triplex vytvořit. Samotná detekce pak probíhá tak, že buňky E. coli vystavíme působení oxidu osmičitého, chloracetaldehydu a psoralenu, izolujeme plasmidy a v nich se snažíme najít báze modifikované působením použitých chemikálií. Porovnáním modifikací, které pozorujeme in vitro s těmi získanými z buňky odhadneme, jestli zde může dojít ke vzniku nějaké neobvyklé struktury. Uvedeným postupem byl vznik H–DNA přímo prokázán a všechny studie s tímto pozorováním spojené

se shodují na vlivu úrovně nadšroubovicového vinutí jakožto hlavního omezujícího faktoru pro vznik takové struktury. Zároveň byl pozorován vliv prostředí, což potvrdilo, že H–DNA vzniká v mírně kyselém prostředí za přítomnosti H^+, protože cytosin vyžaduje protonaci (odtud taky název H–DNA) zatímco *H–DNA preferuje vysoké koncentrace bivalentních kationtů Mg^{2+}.[8] Úroveň nadšroubovicového vinutí závisí na transkripci. RNA polymeráza vytváří domény záporného a kladného nadšroubovicového vinutí, a tak podněcuje vznik H–DNA/*H–DNA, která dále stimuluje rekombinaci. Další jev, který by H-DNA mohla ovlivňovat je replikace, bohužel toto zatím in vivo nebylo dokázáno stejně tak, jako zatím nelze tvrdit, že ovlivňuje strukturu chromatinu, ačkoliv to bylo v genu octomilky hsp26 pozorováno. V této oblasti je zapotřebí další výzkum.[8]

Kapitola 3

Existující přístupy pro detekci triplexů

3.1 Vyhledávání intermolekulárních triplexů

Intermolekulární triplexy mají zejména díky TFO velký význam. TFO mohou však utvořit triplex pouze na určitých vazebných místech označovaných jako cílové sekvence TFO (dále jen TTS — *triplex–forming oligonucleotide target sequence*). Analýzou lidského genomu z UCSC databáze (verze hg12; June 28, 2002), jejíž cílem bylo takové sekvence nalézt, se zabývá (Goni, 2004)[10]. TTS definuje jako polypurinovou sekvenci libovolné velikosti takovou, že při vytvoření triplexu nedojde k žádným neshodám v párování. Aby bylo možné porovnat, jestli množství nalezených TTS odpovídá počtu TTS, které bychom očekávali v náhodné sekvenci, je použit jednoduchý náhodný model, který předpokládá binomické rozdělení TTS. Podle tohoto modelu je možné spočítat očekávané množství P TTS dané délky n v konkrétním genomu délky m. Rovnice 3.1 vyjadřuje pouze aproximaci (dostatečně přesnou) P, která je v (Goni, 2004)[10] odvozena a použita.[10]

$$P = 2 \times n \times (m - n - 1) \times \alpha^2 \times (1 - \alpha)^{n-1} \tag{3.1}$$

V ideálním binomickém rozložení by mělo být $\alpha = 0.5$, ale protože se přechody Pur–Pyr, Pyr–Pur a Pur–Pur v lidském genomu nevyskytují se stejnou pravděpodobností, byla zvolena hodnota $\alpha = 0.44$, která více odpovídá realitě.[10]

Stabilita triplexu odhadovaná s ohledem na jeho teplotu tání závisí na řadě faktorů např. pořadí bází v sekvenci, koncentraci TFO, délce triplexu, přítomnosti modifikovaných nukleotidů v TFO nebo pH. Hrubý odhad teploty tání u paralelních triplexů lze určit pomocí Robertsových a Crothersových rovnic. U antiparalelních triplexů je situace o něco složitější. Teplota tání závisí na koncentraci divalentních kationtů a na možnosti existence jiných alternativních struktur DNA. Nicméně Eritja a kolektiv experimentálně ukázali, že antiparalelní triplexy jsou i v nejhorším případě za neutrálního pH jen o trochu méně stabilní než jejich paralelní protějšky. Hranice teploty tání stabilního triplexu byla pro tuto analýzu stanovena na 50°C. Tato teplota je relativně vysoka a dá se předpokládat, že část z nedetekovaných triplexů by byla za fyziologických podmínek stabilní i s nižší teplotou tání.[10]

Z této analýzy vyplynulo, že počet TTS v lidském genomu je několikanásobně větší než v náhodné sekvenci a tento rozdíl ještě více roste s rostoucí délkou hledaného TTS. Hustota TTS (poměr nukleotidů v TTS k celkovému počtu nukleotidů v sekvenci) je ve

všech chromozomech podobná. Co se nukleotidového složení týká, jsou převážně tvořeny adeniny a dá se říct, že čím je TTS delší, tím více adeninů obsahuje. Jeden z důvodů, proč je adeninu v TTS více, je, že v lidském genomu tvoří adenin 60% veškerých purinů. Nicméně z hlediska tvorby triplexu zde důležitější roli hraje fakt, že adenin je pro jeho vznik výhodnější než guanin. Triplex tak nemusí soupeřit s jinou strukturou, která by zde mohla vzniknout — kvadruplexem.[10]

Ačkoliv výsledky analýzy (Goni, 2004)[10] jsou veřejně dostupné a autoři nabízí na vyžádaní i zdrojové kódy v jazyce C, pro uživatele je rozhodně pohodlnější, když se nemusí probírat cizím kódem a pokoušet se ho překládat. (Gaddis, 2006)[9] představuje webový nástroj k detekci vazebných míst TFO s velkou afinitou — http://spi.mdanderson.org/tfo/. Vyhledávací kritéria jsou v porovnání s (Goni, 2004)[10] odlišná. TTS je charakterizována jako polypurinová oblast s vysokým procentuálním zastoupením guaninu a malým množstvím pyrimidinových inzercí. Tato definice je celkem zajímavá, neboť podle (Goni, 2004)[10] je právě velké zastoupení guaninu nevýhodné, protože v takových oblastech mají tendenci vznikat kvadruplexy. Nicméně veškeré parametry vyhledávání jsou nastavitelné, takže zastoupení guaninu je možno zadat menší. Kromě lidského a myšího genomu nabízí aplikace možnost nahrát vlastní sekvenci ale pouze s omezenými možnostmi vyhledávání. Výsledkem vyhledávání je seznam vazebných míst TFO společně s informacemi o délce sekvence, obsahu guaninu, pozici na chromosomu, genové oblasti, v níž byla sekvence identifikována tj. intron, exon, promotor aj., dále obsahuje odkazy do NCBI a PubMed databází na články potvrzující existenci takových sekvencí a odkazy pro zobrazení nalezených sekvencí v rámci svého chromosomu. Zároveň nabízí možnost pomocí `NCBI BLAST` vyhledat podobné sekvence. Jádro programu je naprogramováno v jazyce Perl.[9]

Délka TTS		**Počet jedinečných výskytů**
Krátká	15–18	67 292 (13%)
Střední	19–24	303 789 (58%)
Dlouhá	25–872	148 890 (29%)
Všechny	15–872	519 971 (100%)

Tabulka 3.1: *Přehled zastoupení jedinečných TTS v referenčním lidském genomu hg18 podle (Jenjaroenpun, 2009). Převzato z [15].*

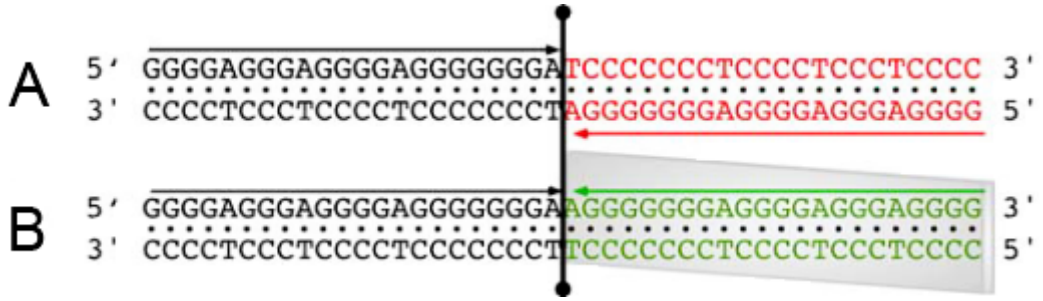
Nejnovější nástroj pro vyhledávání intermolekulárních triplexů představuje (Jenjaroenpun, 2009)[15]. Cílem tohoto nástroje není pouze nalézt místa, kde by mohly vzniknout triplexy, ale i spojit tyto výsledky s dostupnou anotací, která by s nalezeným úsekem mohla být spojena. Program vyhledává polypurinové sekvence, podle zadaných parametrů (pozice na chromosomu, minimální/maximální délka, procentuální zastoupení guaninu, tolerovaný počet vložených pyrimidinů, maskování repetic). Výsledky tohoto vyhledávání jsou následně spojeny s anotací a informacemi o výskytu jiných zvláštních DNA struktur, jako jsou např. CpG ostrůvky nebo kvadruplexy, a společně zobrazeny v UCSC genome browseru. Vyhledávání probíhá na referenčním lidském genomu (hg18) s výchozím nastavením parametrů zvoleným autory aplikace (parametry je možno změnit). Minimální délka 15 bp je odvozena z poznatků, že delší TTS jsou jednak stabilnější a jednak více jedinečné. I když 15 bp ještě nezaručuje velkou jedinečnost, může se už i tak jednat o zajímavou oblast, překrývá-li se

s vazebným místech transkripčního faktoru nebo jiné regulační oblasti.[15]

Analýza lidského genomu podle (Jenjarpoenpu, 2009)[15] vykazuje zajímavé výsledky. Délka TTS koreluje s její jedinečností. TTS délky 15–19bp jsou k nalezení na více místech genomu, zatímco naprostá většina delších TTS je v genomu jedinečná. Přehled jedinečných sekvencí z lidského referenčního genomu hg18 získaných pomocí programu `NCBI BLAST` podává tabulka 3.1. Pyrimidinové inzerce v TTS nezanedbatelně narušují stabilitu případného triplexu. Modifikací TFO však lze tuto překážku částečně překonat, a proto je nutné je uvažovat.[15]

3.2 Detekce intramolekulárních triplexů

Důležitý předpoklad pro možný výskyt alternativní struktury DNA je nějaký motiv nebo vzor v sekvenci. Takovým vzorem mohou být obrácené nebo zrcadlové repetice zachycené na obrázku 3.1. (Schroth, 1995) hledá takové repetice v genomu kvasnic, E. Coli a v lidském genomu. Obrácené repetice jsou jednou z podmínek pro vznik vlásenkové struktury, takže pro nás nejsou moc zajímavé. Na druhou stranu zrcadlové repetice mohou sloužit jako základní stavební kámen pro tvorbu intramolekulárního triplexu. Problém je, že pouze malé procento těchto zrcadlových repetic má potenticál triplex skutečně vytvořit. Na základě faktu, že delší repetice mají potenciál vytvořit stabilnější struktury, zavedli autoři omezení na hledané repetice. Zaměřili se na repetice dlouhé alespoň 10bp (po obou stranách) s možnou mezerou 3–6bp. Minimální požadavek na délku repetice byl odvozen z experimentů, které ukázaly, že oblasti s takto dlouhými repeticemi mají *in vitro* schopnost triplex utvořit. Toto omezení má za příčinu nalezení repetic s opravdu nejvýraznějším potenciálem vytvořit triplex. Kromě nároků na délku jsou rovněž stanovena další dvě přísná kritéria. Repetice musí být čistě homopurinová nebo homopyrimidynová, nesmí obsahovat více jako 80% A–T párů a žádné neshody, ke kterým by mohlo při tvorbě triplexu dojít. S těmito kritérii bylo zjištěno, že v podstatě pouze lidský genom má takové oblasti. Jak v použité DNA sekvenci E. Coli, tak v použité DNA sekvenci kvasnic byla taková oblast nalezena pouze jedna, zatímco v lidské DNA jich bylo nalezeno 22.[19]



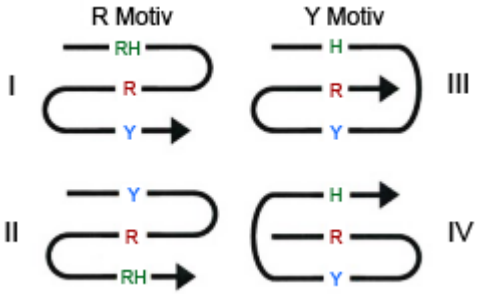
Obrázek 3.1: *A — obrácená repetice. B — zrcadlová repetice. Obrázek převzat z [5].*

(Hoyne, 2000)[11] představuje jiný přístup k vyhledávání intramolekulárních triplexů. Soustředí se na vyhledávání čtyř kanonických typů intrastrand triplexů z obrázku 3.2. Celá analýza je založena na jazyce `Palingol`, který slouží k popisu sekundárních struktur nukleových kyselin a prohledávání databází sekvencí.[4] Veškeré specifikace použité pro tuto analýzu jsou dostupné na vyžádání u autorů. Palingol umožňuje popsat sekundární strukturu pomocí sady vlásenkových struktur a vztahů mezi nimi. Triplex je pak definován jako

dvě vlásenkové struktury, které sdílejí společné homopurinové vlákno. Nutno podotknout, že v tomto kontextu je třeba chápat vlásenkovou strukturu spíše jako sekvenci potenciálně spárovatelných nukleotidů, než rovnou jako stabilní strukturu nukleové kyseliny.[11]

Hledání probíhá ve dvou fázích. V první fázi `Palingol` identifikuje dvě sady vlásenkových struktur. Každá vlásenka z první sady je ohodnocena skórovací maticí a má-li takovou strukturu, aby nukleotidové páry v ní obsažené vyhovovaly Watsonovu-Crickovu párování bází, a zároveň přesahuje minimální zadanou délku, je uchována. Druhá sada vlásenek se liší pouze v tom, že namísto Watsonova-Crickova párování, je vyžadováno Hoogsteenovo nebo reverzní Hoogsteenovo párování. Druhá fáze spočívá v nalezení „Watsonových-Crickových" vlásenek a „Hoogsteenových" vlásenek, které sdílejí homopurinové vlákno. Taková struktura už by mohla mít potenciál vytvořit triplex, nicméně aby ji program definitivně za takovou strukturu označil, musí ověřit, jestli splňuje velmi specifické podmínky. Tyto podmínky byly empiricky zjištěny a spadá mezi ně např. délka těla vlásenek musí být větší než 8 nukleotidů, hlavičky vlásenek musí velikostně spadat do intervalu 0 až 10 nukleotidů, ... Detailně jsou tyto podmínky popsány v (Hoyne, 2000)[11] v kapitole *Materials and Methods*.[11]

Tímto přístupem analyzovali autoři genom E. coli, Synechocystis a Haemophilus influenzae a v každém nalezli více rozptýlených výskytů stejné kopie konkrétní oblasti schopné vytvořit triplex — PIT (potential intrastrand triplex). V každém genomu byly objeveny PIT některé z uvedených tříd z obrázku 3.2. Tyto výsledky mohou být považovány za významné, protože v náhodných sekvencích se stejným poměrem obsahu nukleotidů nalezeny nebyly. Dále, co se statistické významnosti týká, je očekávatelné, že pokud v E. coli a v Synechocystis byly nalezeny PIT třídy II, měly by se zde se stejnou nebo podobnou pravděpodobností nacházet i PIT třídy I, ale tak tomu není. Další fakt, který vyvrací náhodu je, že se v rámci daného organismu jednalo vždy o jednu sekvenci ve více výskytech.[11]



Obrázek 3.2: *Čtyři třídy intrastrand triplexů podle [11]. Šipka naznačuje orientaci vlákna. R, purinové vlákno (červeně); Y, pyrimidinové vlákno (modře); H, vlákno, které se k duplexu váže podle Hoogsteenova párování (zeleně); RH, vlákno, které se k duplexu váže podle reverzního Hoogsteenova párování (zeleně).*

Všechny doposud zmíněné postupy vycházely z detekce homopurinových úseků a kvůli dostatečné stabilitě případného triplexu povolovaly jen minimální přerušení pyrimidiny. Tento přístup je odpovídající, hledáme-li pouze nejpravděpodobnější místa, kde by triplex mohl vzniknout. (Lexa, 2011)[16] přináší nový přístup založený na pracích, které prokazují, že existují i méně „dokonalé" triplexy vykazující více strukturních nedostatků a přitom stále schopné si udržet svoji strukturu. Na základě těchto studií se autoři rozhodli uvažovat následující typy vad, které mohou narušovat stabilitu triplexu:

- K **neshodě v párování bází** dochází při vzniku tripletů, které neumožňují vznik

Typ triplexu	Izomorfní skupina	Triplety
paralelní	a	T*A:T T*G:C C*G:C
	b	G*G:C G*T:A T*C:G
antiparalelní	c	T*A:T A*A:T
	d	A*G:C C*A:T
	e	T*C:G G*G:C

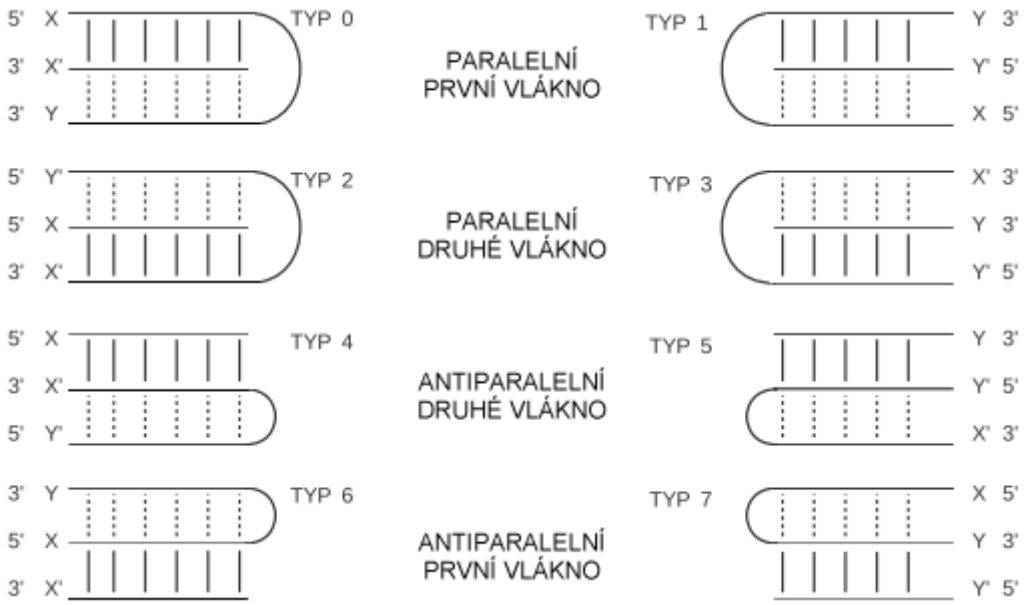
Tabulka 3.2: *Rozdělení tripletů do izomorfních skupin na základě simulace v programu* `AmberTools`. * Hoogsteenova vazba, : Watsonova–Crickova vazba. Převzato z (Lexa, 2011)[16].

silné Hoogsteenovy nebo reverzní Hoogsteenovy vazby. Schopnost vytvářet různě silné vazby je spojena s počtem vodíkových můstků, které mohou být vytvořeny mezi bázemi na druhém a třetím vlákně triplexu. Představovaný algoritmus přiřazuje všem tripletům skóre. Toto skóre odpovídá odhadovanému množství energie, jakým triplet přispívá ke stabilitě triplexu.[16]

- **Geometrická neshoda** vzniká, jestliže sousední triplety nespadají do stejné izomorfní skupiny (rozdělení tripletů do izomorfních skupin je v tabulce 3.2). Tato vada zatěžuje cukrfosfátovou kostru třetího vlákna triplexu a zabraňuje vzniku optimálních vodíkových můstků. V některých případech může kostra v místech neizomorfních tripletů nabývat alternativní klikatou „cikcak“ strukturu podobnou struktuře Z-DNA. V takovém případě není ani při změně izomorfní skupiny triplex nijak penalizován, protože „cikcak“ struktura způsobí uvolnění mimořádné zátěže z kostry.[16]

Mezi další známé faktory ovlivňující vznik triplexu patří sekundární struktura čtvrtého vlákna tj. volného vlákna, které se neúčastní vzniku triplexu; soupeření s ostatními alternativními strukturami, které by v daném místě mohly vzniknout; rozložení C^+ a další vlivy elektrostatických sil. Tyto vlivy algoritmus nebere v úvahu, protože jsou netriviálně závislé na prostředí. Algoritmus prostředí neuvažuje, zkoumá pouze vlivy plynoucí přímo ze sekvence.[16]

Základní myšlenka algoritmu pochází z algoritmu pro vyhledávání palindromů využívajícího metodu dynamického programování k jejich hledání. Vzhledem k tomu, že palindromy mají tendenci vytvářet vlásenkové struktury a podobnost vlásenkových struktur a triplexů byla naznačena výše u metody hledání triplexů pomocí Palingolu, je tento algoritmus dobrým výchozím bodem. Matice dynamického programování (viz obrázek 4.4) je sestrojena tak, aby jedna strana obsahovala prohledávanou sekvenci a druhá strana tu samou sekvenci zapsanou pozpátku. Díky takovému zápisu obsahuje hlavní antidiagonála všechny počáteční pozice třetího vlákna potenciálního triplexu se smyčkou liché délky. Stejně tak sousedící antidiagonála vlevo od hlavní obsahuje všechny počáteční pozice třetího vlákna potenciálního triplexu se smyčkou sudé délky. Při takovémto zápisu označují antidiagonály místa, kde



Obrázek 3.3: *Osm typů detekovaných intramolekulárních triplexů. X a Y jsou nukleotidy ze stejného vlákna, které se účastní tvorby triplexu. Převzato z [16].*

by mohly začínat úseky triplexu, které se dokážou ohnout a navázat ke svému vlastnímu duplexu. Prohledávání začíná z hlavní antidiagonály a její sousední antidiagonály, které jsou předvyplněny nulami. Potenciální triplexy jsou pak reprezentovány diagonálami zde začínajícími.[16]

Nutnost použít metodu dynamického programování plyne z možnosti vkládat do triplexů mezery. Stejně jako při zarovnávání sekvencí můžeme tento problém reprezentovat stromem, který je převoditelný na matici. Oproti algoritmu na hledání palindromů tento algoritmus zavádí několik nových aspektů. Prvním z nich je dříve zmíněné zavedení izomorfních skupin. Dalším novým aspektem je způsob párování. U palindromů stačí uvažovat jednoduché párování A-T, C-G. U triplexů musíme uvažovat jinak. Triplex je sekvence tripletů, které mohou mít různé kombinace nukleotidů. Existuje 16 možných párování pro paralelní DNA vlákna a 16 pro antiparalelní vlákna. Pro všechny tyto páry je v programu předem vypočítaná tabulka s hodnotami skóre.[16]

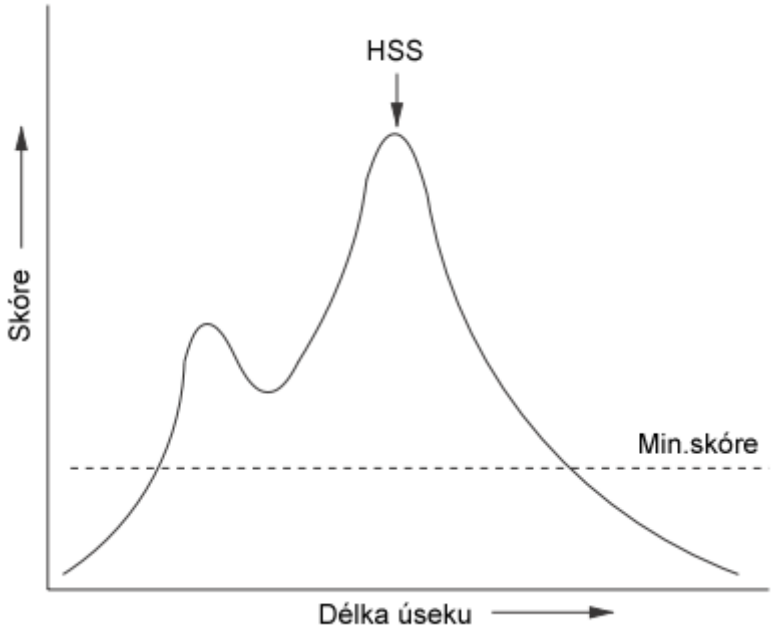
Výpočet v matici začíná na hlavních antidiagonálách, jejichž skóre je na počátku inicializováno na 0. Skóre na každé pozici $[i, j]$ je získáno jako maximum z následujících možností:

- Prodloužení předchozího triplexu podél diagonály — nukleotidy na současné pozici jsou porovnány a pokud tvoří pár obsažený v tripletech z tabulky 3.2, je jim přiřazeno kladné skóre, v opačném případě jsou penalizovány záporným skóre.
- Vložení mezery ve směru původní sekvence (inzerce)
- Vložení mezery ve směru zrcadlové sekvence (delece)

Maximální skóre je uchováno a dosazeno jako výsledek do aktuálního pole matice. Aby do celkového skóre nebylo započítáno volné vlákno a smyčka triplexu, využívá algoritmus kombinaci lokálního a globálního zarovnání. U prvních $2l$ antidiagonál, kde l je nastavitelný parametr udávající maximální velikost smyčky, je skóre v duchu lokálního zarovnání

upraveno vždy tak, aby nekleslo pod nulu. Po $2l$ antidiagonálách pokračuje výpočet podle globálního zarovnávání, takže skóre už pod nulu klidně klesnout může.[16]

Nejlepší triplexy lze v matici identifikovat podle nejvyššího skóre. Algoritmus úseky s vysokým skóre (*HSS — high scoring segment*) identifikuje už v průběhu a to podobnou technikou jakou využívá program BLAST. Jakmile v některé části matice stoupne skóre nad zvolenou hranici, program tuto část označí jako začátek potenciálního triplexu. Skóre je od této chvíle monitorováno až do doby, než zase pod zvolenou hranici klesne (viz obrázek 3.4). Úsek od první označené antidiagonály až po antidiagonálu, ve které bylo skóre nejvyšší, je označen jako potenciální triplex.[16]



Obrázek 3.4: *Detekce HSS. Převzato z [16].*

Časová složitost výpočtu celé poloviny matice je $n^2/2$, nicméně pravděpodobnost nalezení potenciálního triplexu klesá s jeho požadovanou délkou. Ve většině případů dokonce stačí vypočítat pouze $2l$ diagonál, kde l označuje maximální délku triplexu. Časová náročnost spíše tedy odpovídá $O(2ln)$. Co se prostorové složitosti týká, vyžaduje algoritmus k výpočtu pouze hodnoty z dvou předchozích antidiagonál, tudíž je jeho prostorová složitost $O(2n)$.[16]

Platnost výsledků algoritmu byla ověřena několika testy na sadě sekvencí. Každá sekvence má délku podobnou délce genomu E.coli tzn. $\sim 4,7$ milionů bází. Tato sada obsahuje náhodnou sekvenci, kompletní genom E. coli (verze U0096.1 i U0092.2), náhodně zpřeházenou sekvenci E. coli, část sekvence z lidského chromozomu 5 a její náhodně zpřeházenou verzi. Z těchto testů vyplynulo, že náhodné sekvence mají nižší zastoupení HSS. Lidské sekvence jsou v porovnání se sekvencí E. coli, do které byla na každých 10 000 nukleotidů přidána triplexové sekvence, na výskyt těchto oblastí bohatší. Další test spočívá v porovnání výsledků algoritmu s databází potenciálních triplexů z (Cer, 2010)[6]. Algoritmus přednostně nachází výsledky obsažené v této databázi a navíc se mu podařilo detekovat i dvě zajímavé oblasti, které podle (Bacolla, 1991)[2] a (Becker, 1998)[3] mají schopnost

vytvořit alternativní triplexovou strukturu.[16]

Alternativní způsob ověření správnosti výsledků je založen na porovnání s (Hoyne, 2000)[11]. (Hoyne, 2000)[11] sice hledá intrastrand triplexy, ale každý intrastrand triplex je složen ze tří po sobě jdoucích sekvencí, z nichž dvě po sobě jdoucí mohou stačit pro tvorbu intermolekulárního triplexu. Se správným nastavením parametrů tento test potvrdil výskyt všech 25 oblastí z (Hoyne, 2000)[11].[16]

Kapitola 4

Návrh a implementace algoritmu

4.1 Volba přístupu

Z metod popsaných v předchozí kapitole je z praktického hlediska nejpokročilejší algoritmus prezentovaný v (Lexa, 2011)[16]. Kromě toho, že tento agoritmus při hledání uvažuje pokročilé podmínky, které ostatní algoritmy neuvažují, je zároveň i efektivně implementován, má dobrou časovou i paměťovou složitost a jeho zdrojové kódy v jazyce C jsou veřejně dostupné. Z těchto důvodů byl zvolen jako výchozí algoritmus pro tuto práci.

Na druhou stranu má zvolený algoritmus i své nedostatky. Ačkoliv zpracovává vstupní sekvence ve FASTA formátu, ze znaků, které FASTA formát povoluje, zahrnuje do výpočtu pouze znaky reprezentující konkrétní nukleotidy tzn. `A, C, G, T`. Zbylé povolené znaky nedokáže zpracovat. Další jeho nedostatek spočívá v absenci zpětného zarovnání. To má za následek, že algoritmus nedokáže vypsat přesnou sekvenci nalezených triplexů, které obsahují inzerce resp. nedokáže označit, kde se inzerce nachází. Zbylé nedostatky souvisí už více s celým programem, než pouze s algoritmem samotným. Z uživatelského hlediska je vždy příjemnější aplikace s grafickým uživatelským rozhraním, než konzolová aplikace. Program grafické uživatelské rozhraní v podobě webové aplikace měl, to v současné době ale už není funkční.

Na řešení zmíněných nedostatků je tato práce zaměřena. Navíc kromě jejich řešení představuje v rámci vytvořené webové aplikace i zcela novou vlastnost — schematickou vizualizaci triplexů.

4.2 Jádro programu

Zatímco hlavní myšlenka algoritmu zůstává nezměněna a funguje stejně, jak je popsáno na konci kapitoly 3.2, zbylá část programu prošla výraznými změnami. Originální verze programu dostupná na `http://www.fi.muni.cz/~lexa/triplex/` není dostatečně přenositelná a na 64-bitovém systému nejde přeložit. Bohužel server, pro který byla celá komplexní aplikace vyvíjena, 64-bitový je. Jádro programu tedy tvoří o něco vývojově starší verze, které ale bohužel chybí některé důležité vlastnosti z nejnovější verze. Základními cíli pro úpravy programu bylo znovu zavést tyto vlastnosti:

- podpora načítání souborů ve FASTA formátu
- zpracování vstupu po částech, aby u velkých vstupů nenastával nedostatek paměti při alokaci antidiagonál

- úprava algoritmu tak, aby bral v úvahu změnu izomorfní skupiny

Pro načítání FASTA souborů jsem použil *FASTA/FASTQ Parser*, malou knihovnu pro jazyk C určenou k parsování souborů ve FASTA a FASTQ formátu.[1]. Knihovna pracuje nad proudem dat, lze ji tedy kromě souborů použít i pro parsování jinak načtených řetězců. Dalším užitečným rysem je schopnost knihovny zpracovat i komprimované soubory, což může mít nemalý přínos pro vytvořenou webovou aplikaci. Například komprimovaná sekvence E. coli má místo 4,48 MB pouhých 1,36 MB, chromozom 1 lidského genomu lze zmenšit z 242 MB na 70,3 MB. Načtené sekvence uchovává knihovna v následující struktuře:

```
typedef struct {
    kstring_t name, comment, seq, qual;
    int last_char;
    kstream_t *f;
} kseq_t;
```

Name, comment a qual (quality) jsou údaje o sekvenci načtené z FASTA hlavičky. Seq reprezentuje samotnou načtenou sekvenci. Zbylé proměnné ze struktury `kseq_t` používá knihovna pro své interní potřeby.[17] Program zpracuje ze vstupních dat pouze první FASTA sekvenci, ostatní ignoruje. Toto chování jsem zvolil s ohledem na roli tohoto programu ve vytvořené webové aplikaci.

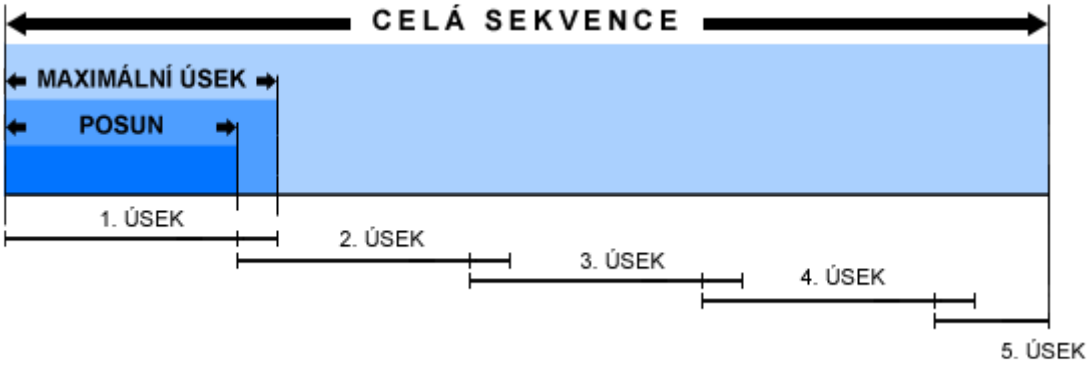
Program má definovanouu maximální povolenou velikost prohledávané sekvence. Maximální velikost jde měnit pomocí parametru `-g`, a tak postupně dojít až k hraničnímu množství paměti, kterou má program povolenou alokovat. Pokud je načtená sekvence delší než toto maximum, je prohledávána po částech. Dělení sekvence nevyžaduje žádnou paměť navíc. Pro každý segment je ale nutné znovu inicializovat pole antidiagonál, to má mírný vliv na výkon algoritmu. Dělení je implementováno posouváním ukazatele v načtené sekvenci. Menší problém nastává v bodech, kde je sekvence rozdělována. Pokud by zde byla oblast potenciálního trpiplexu, nebyla by nalezena. Proto se ukazatel neposunuje v sekvenci o maximální povolenou velikost, ale uvažuje určitý překryv mezi jednotlivými úseky. Překryv o je spočítán jako:

$$o = l_{max} + steps \tag{4.1}$$

kde l_{max} je maximální velikost smyčky triplexu a *steps* je počet kroků algoritmu. Tato myšlenka je zobrazena na obrázku 4.1. Překryvy rovněž znamenají, že překrývané části budou prohledávány dvakrát, takže může dojít k nalezení více výsledků, než kdyby sekvence nebyla rozdělena. Délka běžného překryvu je v porovnání s délkou sekvence zanedbatelná, takže dvojí prohledávání překryvů má minimální vliv na rychlost.

Další změnu podstoupila klíčová funkce `get_max_score(...)`, která obstarává výpočet nejlepšího skóre pro současnou pozici v matici. Funkce byla doplněna o důležitou podmínku, která v případě schody na diagonále rozhoduje, jestli odečíst penalizaci za změnu izomorfní skupiny nebo přičíst bonus za její dodržení. Tato podmínka testuje, zda triplet spadá do stejné izomorfní skupiny a pokud ne, tak zjišťuje, jestli zde může vznikat „cikcak“ strutura páteře, tak jak je podle (Lexa, 2011))[16] popsáno na konci předchozí kapitoly. Celá podmínka je převzata ze zdrojových kódů nabízených jako doplňující materiál k (Lexa, 2011)[16].

[1] Tato knihovna je volně dostupná na `http://lh3lh3.users.sourceforge.net/parsefastq.shtml` pod MIT licencí



Obrázek 4.1: *Překryv při rozdělování sekvence na části. Velikost maximálního úseku je na obrázku zvolena jako čtvrtina délky celé sekvence. Přesto bude sekvence kvůli překryvům rozdělena na pět částí.*

Program má pouze jeden povinný parametr, tím je cesta k souboru se vstupní sekvencí. Dalších šestnáct parametrů je nepovinných a v případě, že nejsou zadány použije program přednastavenou výchozí hodnotu, jinak jimi lze ovlivnit veškeré parametry hledání včetně všech penalizací. Popis jednotlivých parametrů obsahuje nápověda programu, tu lze vyvolat spuštěním programu s parametrem `-h`.

4.3 Zpracování všech povolených znaků FASTA formátu pro nukleové kyseliny

FASTA formát nabízí mimo znaků A, C, G, T pro nukleotidy i další znaky, které mohou zastupovat více nukleotidů najednou. Původní program zpracování těchto symbolů neuvažuje. V rámci této práce jsem ho rozšířil tak, aby speciální znaky z tabulky 4.1 dokázal zpracovat.

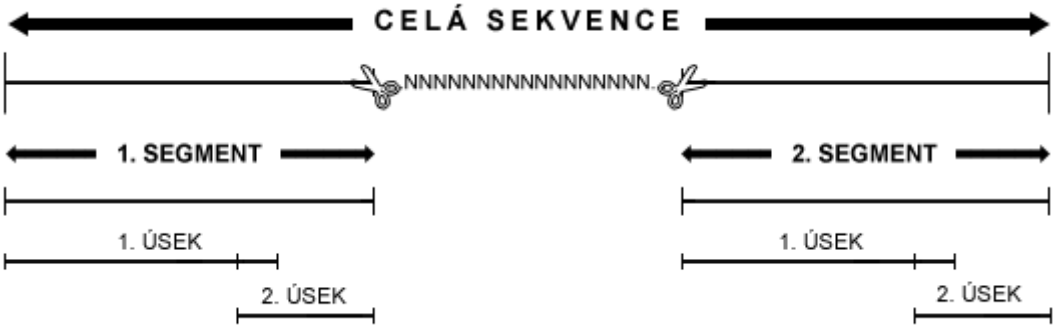
R	purin	**W**	slabě interagující
Y	pyrimidin	**B**	jiný než A
K	báze s ketoskupinou	**D**	jiný než C
M	báze s aminoskupinou	**H**	jiný než G
S	silně interagující	**V**	jiný než T
N	libovolná báze	**-**	mezera

Tabulka 4.1: *Speciální znaky povolené v souborech ve FASTA formátu.*

K nahrazování speciálních znaků dochází ve chvíli, kdy alespoň jeden ze znaků předaných do funkce `get_max_score(...)` spadá mezi speciální znaky. Tyto znaky jsou rozpoznány a je zavolána funkce `handle_special(...)`, ve které jsou za znaky dosazeny nejvhodnější nukleotidy. V programu je pevně definováno pole reprezentující tabulku 4.1, které každému znaku přiřazuje vlastní množinu nukleotidů. Výběr nejlepší kombinace probíhá tak, že nad množinami nukleotidů, které znaky zastupují, je proveden kartézský součin (obrázek 6.2.

Z něj je vybrána dvojice, která pokud možno splňuje dříve zmíněnou podmínku izomorfismu a má podle skórovacích tabulek nejvyšší skóre. Ačkoliv zkoušet všechny variace není nejelegantnější metoda, v tomto případě je její použití adekvátní. Maximální počet možností, který zde může nastat je devět v případě, že se jedná o libovolnou dvojici ze znaků B, D, H, V (každý z nich zastupuje tři nukleotidy).

Obrázek 4.2: *Zpracování speciálních znaků FASTA formátu. Nad množinami, které zastupují vstupní znaky R a Y je proveden kartézský součin a z něj je vybrána dvojice s nejlepším skóre. Tu bez kontextu prohledávání nelze určit. K jejímu určení je zapotřebí znát typ hledaného triplexu a aktuální izomorfní skupinu.*



Obrázek 4.3: *Kompletní mechanismus dělení sekvence na menší části. Ukázková sekvence obsahuje oblast znaků* `N`*. Nejprve je sekvence „rozstřižena“ podle mezer na dva segmenty (mezerou rozumíme symboly* `N` *a* `-`*). Tyto segmenty jsou dále děleny na úseky tak, jako vysvětluje obrázek 4.1.*

Zpracování znaku `N` je odlišné od ostatních. `N` může zastupovat libovolnou bázi. V reálných sekvencích se zpravidla objevuje na místech, která se nepodařilo nasekvenovat (často kvůli velkému počtu repetic). `N` se tedy nevyskytuje samostatně, ale zpravidla vyplňuje celé delší oblasti. V těchto místech by si algoritmus mohl dosadit zcela libovolné báze tak,

aby vytvořily ideální triplex. Něco takového nemá z praktického hlediska smysl, proto jsou oblasti obsahující `N` interpretovány stejně jako mezery `-`. Sekvence je pak podle mezer rozdělena na menší segmenty. Každý segment musí být zpracován zvlášť téměř jako by byl samostatná sekvence, tyto segment nelze dát dohromady a brát jako jednu sekvenci, protože segmenty před mezerou a za mezerou fyzicky nenavazují. Segmenty jsou dále rozděleny podle maximální povolené velikosti způsobem popsaným v kapitole 4.2. Kompletní princip dělení je znázorněn na obrázku 4.3.

K rozdělení sekvence na segmenty je použita funkce `strtok(...)` ze standardní knihovny `string.h`. Takový způsob dělení nemá žádné dodatečné paměťové nároky, ale vyžaduje navíc jeden sekvenční průchod celou sekvencí.

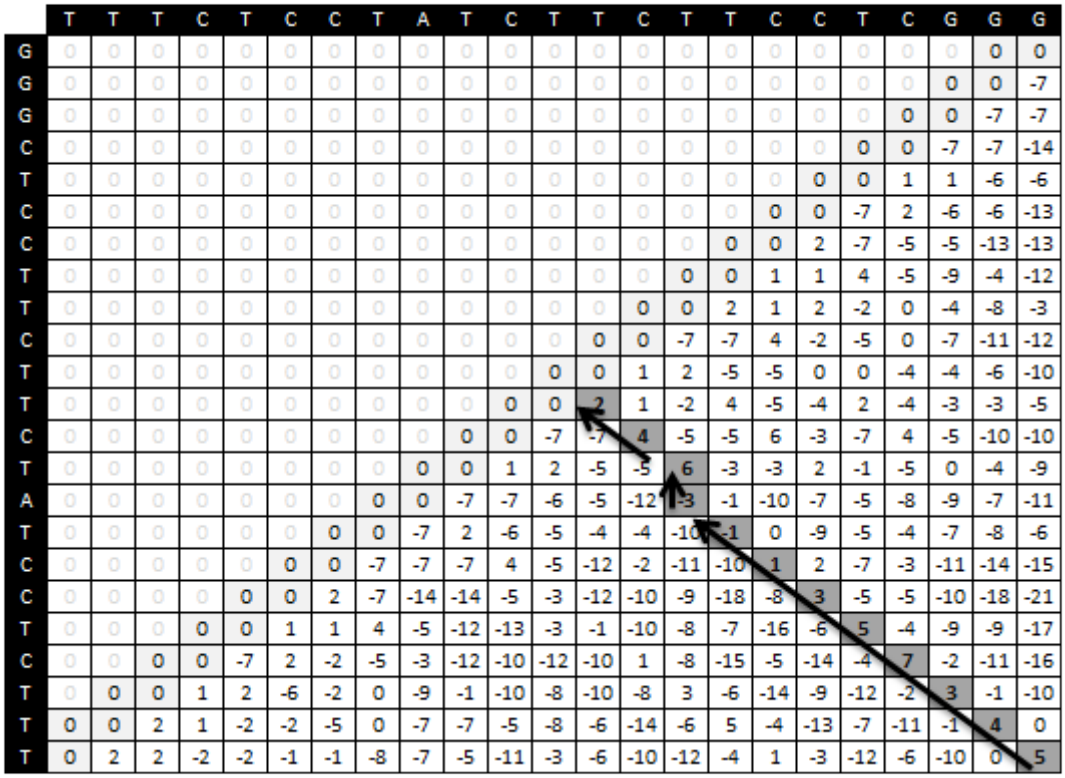
4.4 Úpravy pro zpětný průchod maticí

Jak už bylo řečeno, původní program nesestavuje celou matici dynamického programování, protože ta má kvadratickou paměťovou závislost na délce vstupní sekvence. Pro představu, i kdybychom pominuli pokročilé aspekty tohoto algoritmu, do kterých spadají třeba změny izomorfní skupiny, musíme na každé pozici matice uchovat alespoň skóre a směr, odkud jsme se na tuto pozici dostali. Pro tento účel by nám stačily 2 bajty. Už i na ukázkovém genomu E. coli, který je dlouhý 4 639 250 nukleotidů, dostáváme enormní paměťové nároky $2 \times 4\,639\,250^2\,\mathrm{B} \approx 39,15\,\mathrm{TB}$.

Upravená verze programu je určena pro krátké sekvence, u kterých je paměťově únosné sestavit celou matici. Hlavní algoritmus podstoupil jen malé změny. V průběhu hlavního výpočtu nyní nevypisuje žádné nalezené triplexy, místo toho postupně doplňuje informace do předem alokované matice. V ostatních aspektech zůstává algoritmus nezměněn. Takto vyplněnou matici lze následně použít pro zpětný průchod a tím zjistit přesnou sekvenci triplexu včetně inzercí. Výchozím výstupem programu je tedy jediný triplex získaný zpětným průchodem sestavené matice. Z programu je však použitím nepovinného parametru `-g` možno získat i celou matici v csv formátu. Z této exportované matice je sestaven i obrázek 4.4, který demonstruje činnost programu. Pro tuto demonstraci jsem použil stejnou sekvenci, která je použita v (Lexa, 2011)[16].

Průchod začíná v pravém dolním rohu matice. Po každém kroce se algoritmus posune na novou pozici. Směr posunu je dán pravidlem na současné pozici, skóre na dané pozici je v tuto chvíli už nepodstatné. Průchod probíhá, dokud se algoritmus neposune až k jedné z hlavních antidiagonál nebo dokud nenarazí na nevyplněné pole matice. Ta mohou v dolní polovině matice vzniknout, když algoritmus na základě parametru *min_loop* pro minimální velikost smyčky přeskakuje prvních *min_loop* antidiagonal. Během průchodu program vytváří výstupní sekvenci tak, aby výstupní formát splňoval následující pravidla:

- Párující se nukleotidy jsou vypsány malými písmeny.
- Nepárující se nukleotidy jsou vypsány velkými písmeny.
- Inzerce jsou ohraničeny závorkami.
- Smyčka je od těla oddělena pomlčkami.



	T	T	T	C	T	C	C	T	A	T	C	T	T	C	T	T	C	C	T	C	G	G	G
G	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
G	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-7
G	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-7	-7
C	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-7	-7	-14
T	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	-6	-6
C	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-7	2	-6	-6	-13
C	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	-7	-5	-5	-13	-13
T	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	4	-5	-9	-4	-12
T	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	1	2	-2	0	-4	-8	-3
C	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-7	-7	4	-2	-5	0	-7	-11	-12
T	0	0	0	0	0	0	0	0	0	0	0	0	0	1	2	-5	-5	0	0	-4	-4	-6	-10
T	0	0	0	0	0	0	0	0	0	0	0	0	2	1	-2	4	-5	-4	2	-4	-3	-3	-5
C	0	0	0	0	0	0	0	0	0	0	0	-7	-7	4	-5	-5	6	-3	-7	4	-5	-10	-10
T	0	0	0	0	0	0	0	0	0	0	1	2	-5	-5	6	-3	-3	2	-1	-5	0	-4	-9
A	0	0	0	0	0	0	0	0	0	-7	-7	-6	-5	-12	-3	-1	-10	-7	-5	-8	-9	-7	-11
T	0	0	0	0	0	0	0	0	-7	2	-6	-5	-4	-4	-10	-1	0	-9	-5	-4	-7	-8	-6
C	0	0	0	0	0	0	0	-7	-7	-7	4	-5	-12	-2	-11	-10	1	2	-7	-3	-11	-14	-15
C	0	0	0	0	0	0	2	-7	-14	-14	-5	-3	-12	-10	-9	-18	-8	3	-5	-5	-10	-18	-21
T	0	0	0	0	0	1	1	4	-5	-12	-13	-3	-1	-10	-8	-7	-16	-6	5	-4	-9	-9	-17
C	0	0	0	0	-7	2	-2	-5	-3	-12	-10	-12	-10	1	-8	-15	-5	-14	-4	7	-2	-11	-16
T	0	0	0	1	2	-6	-2	0	-9	-1	-10	-8	-10	-8	3	-6	-14	-9	-12	-2	3	-1	-10
T	0	0	2	1	-2	-2	-5	0	-7	-7	-5	-8	-6	-14	-6	5	-4	-13	-7	-11	-1	4	0
T	0	2	2	-2	-2	-1	-1	-8	-7	-5	-11	-3	-6	-10	-12	-4	1	-3	-12	-6	-10	0	5

Obrázek 4.4: *Zpětný průchod maticí sestavenou během výpočtu. Horní polovina matice nad antidiagonálou je pro výpočet nepodstatná. Šipka znázorňuje směr zpětného průchodu.*

Pro ukázkový průchod z obrázku 4.4 dostáváme následující výstup:

```
gggctccttct--tct(a)tcctcttt
```

Tento řetězec je dále použitelný pro schematické vykreslení triplexu. Vykreslováním se blíže zabývá až kapitola 5.3.

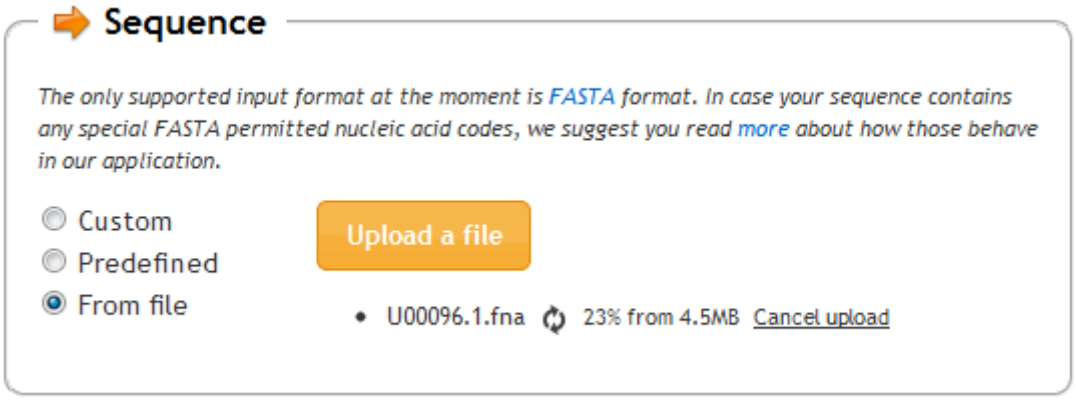
Kapitola 5

Webová aplikace

5.1 Hlavní aplikace

Webová aplikace nabízí pohodlné webové uživatelské rozhraní. Byla vytvořena s cílem zjednodušit a zpříjemnit uživatelům používání vytvořených programů. Jakožto webová aplikace má navíc mnohem větší potenciál. Vyhledané triplexy jsou rovnou schematicky vizualizovány a u předdefinovaných sekvencí nabízí aplikace odkaz do UCSC genome browseru směřující na vyhledanou pozici triplexu v rámci své sekvence. Celá aplikace v anglickém jazyce běží na serveru `http://bioware.fit.vutbr.cz/~zruna/DIP/`.

Serverová část aplikace je vytvořena v jazyce PHP s pomocí českého frameworku Nette. To znamená, že využívá softwarovou architekturu model-view-controller, která striktně odděluje práci s daty, zobrazování dat a zpracování požadavků. O databázovou část aplikace se stará databázový systém MySQL. Klientská část využívá javascriptovou knihovnu jQuery, jQueryUI a zapojuje několik volně šiřitelných pluginů na těchto knihovnách postavených, aby uživatelům nabídla co nejpohodlnější a zároveň moderní uživatelské rozhraní.



Obrázek 5.1: *Výběr vstupní sekvence na webovém rozhraní*

Vstupní formulář pro vyhledávání umožňuje nastavit v podstatě veškeré parametry, které konzolová aplikace používá. Parametry jsou rozděleny do několika kategorií. První kategorie je vstupní sekvence. Krátký popis u této kategorie uživatele informuje o podporovaném formátu vstupní sekvence (FASTA formát) a o tom, jak se aplikace chová, když narazí na některý ze speciálních znaků podporovaných ve FASTA formátu. Sekvenci je

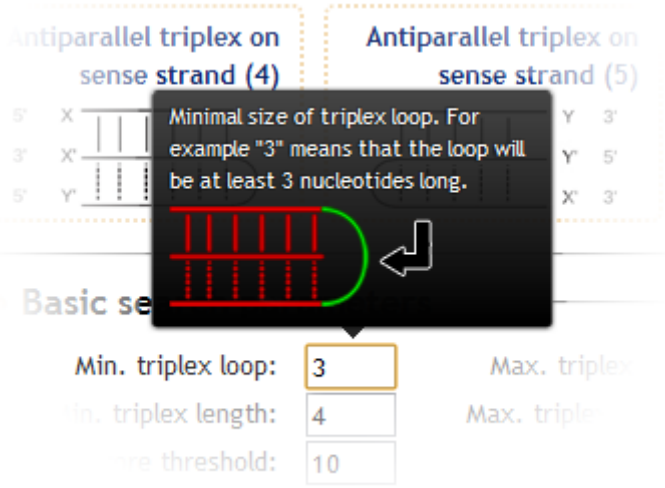
možné zvolit z několika zdrojů:

Vlastní vstup Vstupní sekvence musí být ve FASTA formátu zadána přímo do vstupního pole v prohlížeči.

Předdefinovaný vstup Několik sekvencí je uloženo v přímo na serveru. Jedná se o všechny chromozomy lidského genomu a kompletní genom bakterie Escherichia coli. Z těchto sekvencí může uživatel libovolnou rovnou použít.

Vstup ze souboru Poslední možnost představuje nahrání vlastního souboru. Velikost tohoto souboru je z kapacitních důvodů omezena na 5 MB. Nahrávání souboru na server probíhá asynchronně ihned po zvolení souboru (tedy ne až po odeslání formuláře) a je možné ho v průběhu kdykoliv přerušit (ukázka na obrázku 5.1). Tento způsob má svá pro i proti. Hlavní výhodu je možnost už během nahrávání souboru vyplňovat zbývající část formuláře (ne náhodou je tedy volba vstupní sekvence hned první parametr ve formuláři). Naproti tomu hlavní nevýhoda spočívá v tom, že veškeré nahrané soubory včetně těch nakonec nepoužitých zůstávají uloženy na serveru až do chvíle, než je někdo smaže. Tomuto problému by šlo zamezit automatickým pravidelným spouštěním skriptu, který by adresář s nahranými sekvencemi kontroloval a promazával. K tomuto účelu slouží softwaroví démoni jako např. Cron.

Další kategorii tvoří typ triplexu. Každý typ triplexu je ilustrován obrázkem, aby bylo zřejmé o jakou variantu se jedná. Obrázky jednotlivých typů triplexů jsou přejaty z (Lexa, 2011)[16] a k vidění jsou na obrázku 3.3. Třetí a čtvrtá kategorie představuje základní a pokročilé parametry algoritmu. Každý parametr je opatřen nápovědou, která by měla uživateli objasnit jeho smysl (obrázek 5.2).



Obrázek 5.2: *Ukázka nápovědy jednoho z parametrů.*

Jakmile uživatel odešle vyplněný formulář, je na server odeslán asynchronní požadavek. Uživateli zůstane stránka bez obnovení otevřená, odeslání formuláře ale vytvoří novou záložku (uvnitř aplikace nikoliv prohlížeče) a automaticky do ní uživatele přepne. Každé hledání má svoji záložku, která informuje uživatele o stavu hledání. V okamžik, kdy je

hledání ukončeno, se zde objeví odkaz na výsledky hledání nebo případně se výsledky rovnou samy po dokončení hledání otevřou. Výhodou je, že formulář zůstane i po potvrzení vyplněn stejnými údaji (včetně souboru nahraného na server), a tak uživatel může obratem spustit nové hledání se stejnými parametry pouze se změněným typem triplexu. Další dění na serveru je zachyceno na obrázku 5.3.

Obrázek 5.3: *Tento diagram zobrazuje dění na serveru po odeslání formuláře uživatelem. Detaily jednotlivých kroků jsou rozebrány níže.*

Krok 1 — Zpracování parametrů Validace správnosti parametrů probíhá jak na straně klienta tak na serveru. Vzhledem k tomu, že přijaté hodnoty jsou použity dál jako parametry pro konzolovou aplikaci, je z hlediska bezpečnosti také nutné je „escapovat“, aby nedošlo ke zneužití.

Krok 2 — Spuštění hlavní aplikace S ošetřenými parametry server spustí hlavní konzolovou aplikaci na hledání triplexů a její výstup přesměruje do souboru. Přesměrování je nutné, protože PHP skript dále pracuje s výstupem aplikace, jenomže paměť, kterou smí alokovat, je v současné době limitována nastavením serveru na 128 MB. Přesměrováním do souboru toto omezení obejdeme.

Krok 3 — Vložení do databáze Soubor s výsledky je po 1 000 záznamech načítán a vkládán do databáze. Po vložení do databáze server soubor automaticky smaže.

Krok 4 — Filtrování výsledků Cílem filtrování je zbavit se triplexů, které jsou podmnožinou jiného triplexu a přitom jsou méně kvalitní tak, jak demonstruje obrázek 5.4. Filtrování musí být kvůli paměťovým omezením prováděno po částech. Původní záměr byl filtrovat výsledky rovnou v databázi jediným SQL dotazem. Tento jednoduchý způsob, ačkoliv byl funkční, byl u většího množství výsledků nepoužitelný, protože vykonání filtrovacího SQL dotazu trvalo moc dlouhou dobu, během které byla tabulka uzamčena pro jakékoliv další dotazy a to mělo v podstatě za následek

nepoužitelnost celé aplikace během filtrování. Nakonec tedy bylo zvoleno programové filtrování v PHP skriptu. Skript načítá po dávkách triplexy z databáze seřazené podle jejich začátku a konce. Seřazení probíhá přímo v databázi, která ho díky indexaci sloupců provádí velmi efektivně a rychle. Každý triplex je porovnáván se svými následníky, dokud to má smysl tzn. dokud je konec následujícího triplexu menší nebo roven konci současného triplexu. Naivní přístup k filtrování, který by porovnával každý triplex s každým by měl kvadratickou časovou složitost. Metoda zvolená v této aplikaci má díky omezení maximální a minimální délky triplexu zadanými parametry lepší časovou složitost. Je-li maximální délka triplexu max a minimální min, pak maximální počet porovnání p, které bude muset pro každý záznam udělat je:

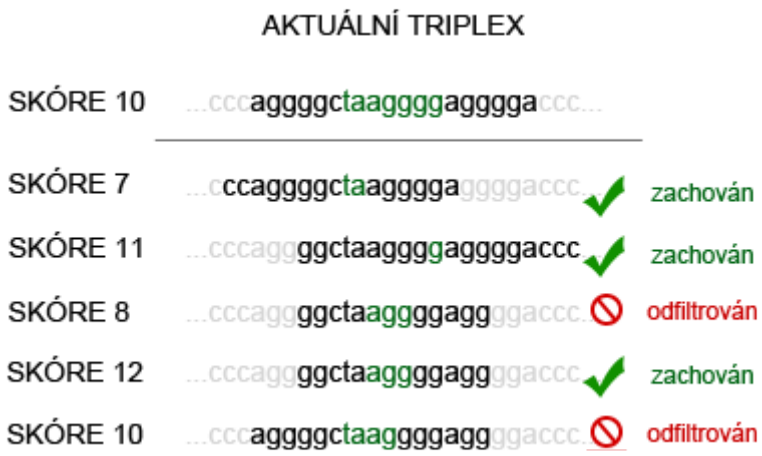
$$p = \sum_{i=0}^{max-min} (max - min + 1 - i) \tag{5.1}$$

Maximální počet porovnání použijeme pro výpočet průměrné časové složitosti.

$$T(n) = n \times p = \Theta(n) \tag{5.2}$$

V nejlepším případě, kdy neexistuje žádný triplex, který je podmnožinou jiného dostáváme lineární časovou složitost

$$T(n) = \Omega(n) \tag{5.3}$$



Obrázek 5.4: *Cílem filtrování výsledků je odstranit triplexy, které jsou podmnožinou jiných triplexů a zároveň jsou méně kvalitní. Podmnožinou triplexu rozumíme sekvenci od počátku triplexu (včetně) po jeho konec (včetně). V aplikaci by k porovnání s prvním ani druhým z triplexů ani nedošlo, protože ty nejsou podmnožinou aktuálního triplexu.*

Krok 5 — Stránkování a výběr výsledků Množství výsledků může dosahovat velkých hodnot, proto aplikace uživatelům nabízí stránkovaní, filtrování a řazení podle jejich potřeby. Výchozí řazení je sestupné podle skóre triplexu.

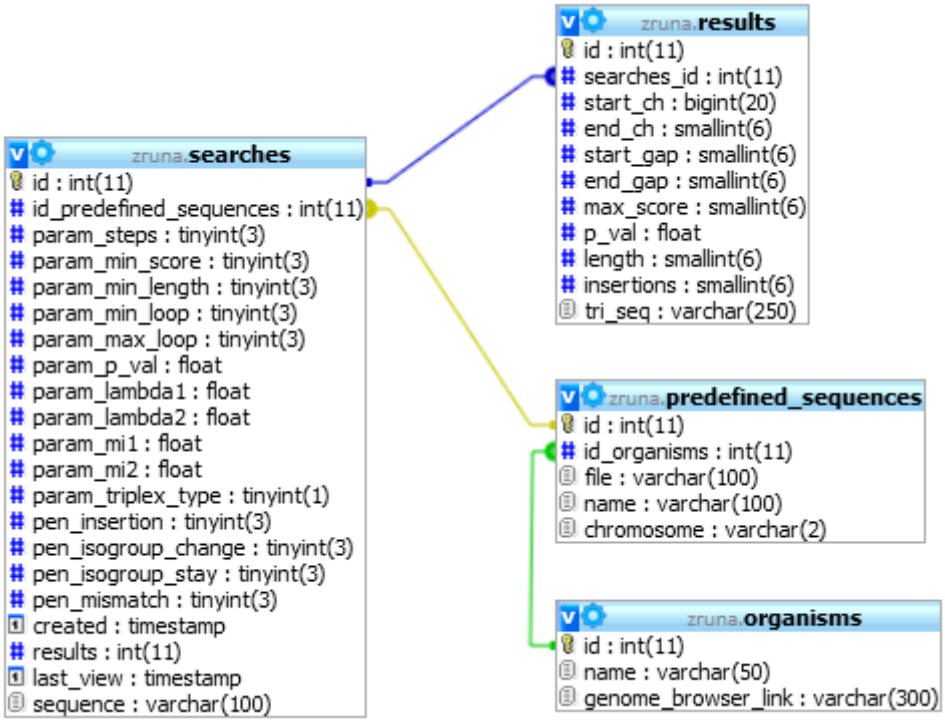
Krok 6 — Zpětný průchod maticí Jakmile aplikace vybere triplexy, které teď bude zapotřebí zobrazit, spustí pro každý z nich modifikovanou verzi programu. Program sestaví matici, provede zpětný průchod a vytiskne výsledek.

Krok 7 — Vizualizace triplexů Výstupy obdržené v kroku 6 jsou použity pro vizualizaci triplexu. Více o vizualizaci v kapitole 5.3.

Krok 8 — Zobrazení výsledků Jednotlivé triplexy jsou zobrazeny v tabulce, která obsahuje veškeré informace o konkrétním triplexu včetně náhledu. Výsledky lze exportovat do gff3 souboru.

5.2 Databáze

Databáze je tvořena čtyřmi tabulkami tak, jak ukazuje obrázek 5.5.



Obrázek 5.5: *Schéma použité databáze.*

Tabulka searches Obsahuje záznamy o každém provedeném vyhledávání. Ukládá použité parametry, cestu k souboru se sekvencí, datum posledního zobrazení, cizí klíč do tabulky predefined_sequences aj. I když se může zdát nadbytečné, že tabulka uchovává i počet vyhledaných triplexů, je to z hlediska výkonnosti aplikace a zatížení databáze přínosné. Výsledků mohou být klidně i statisíce a zjišťovat jejich počet pomocí agregační funkce COUNT trvá v takových případech v řádech sekund. Kdežto pokud obětujeme čtyři bajty navíc pro uložení počtu výsledků, dokážeme jejich počet zjistit vždy za pár milisekund.

Atribut last_view je aktualizován vždy, když jsou výsledky zobrazeny. V současné době není nijak dál využit, ale do budoucna by mohl sloužit při údržbě databáze k identifikaci nepoužívaných výsledků, které by byly následně smazány.

Tabulka results Každý záznam této tabulky reprezentuje jeden nalezený triplex. Záznam obsahuje informace o kvalitě, umístění triplexu, jeho sekvenci před zpětným zarovnáním a cizí klíč do tabulky searches.

Tabulka predefined_sequences Uchovává seznam předdefinovaných sekvencí a informaci o tom, kterého organismu se týkají.

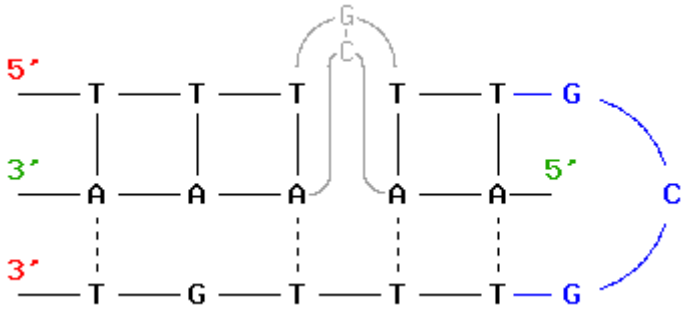
Tabulka organisms Obsahuje názvy organismů pro předdefinované sekvence a odkaz do UCSC genome browseru.

5.3 Vykreslování triplexů

Vykreslování triplexů je implementováno v jazyce PHP. Slouží k němu třída `TriVis`. Není nijak pevně vázaná na tuto webovou aplikaci, byla vytvořena tak, aby bylo snadné ji použít i mimo ni. Objekty třídy `TriVis` vyžadují dva parametry v konstruktoru. Prvním z nich je vykreslovaná sekvence, která musí splňovat specifický formát (viz dále), a druhým je typ triplexu. Typy triplexů odpovídají typům z (Lexa, 2011)[16] a i výsledná podoba obrázků je obrázky z této práce zásadně ovlivněna.

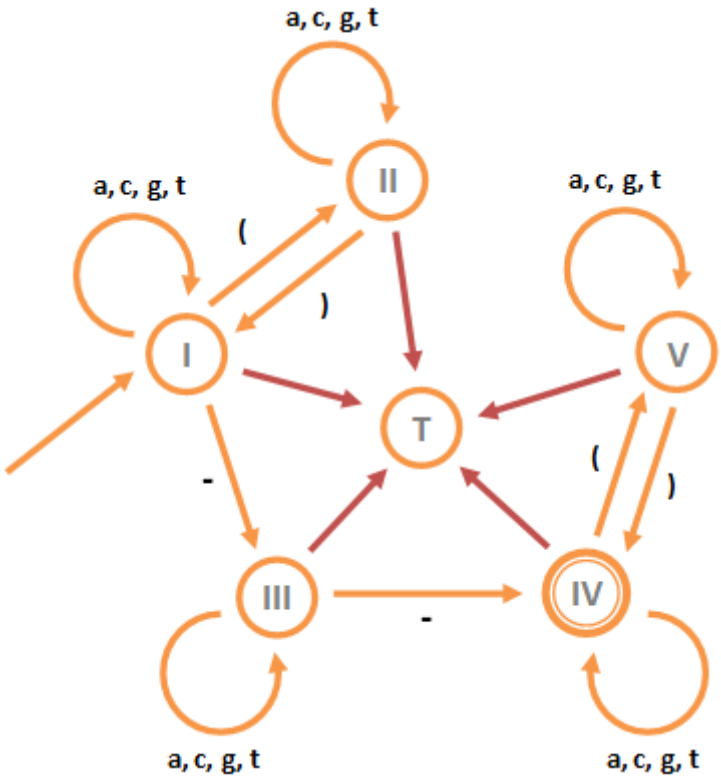
Pro tvorbu obrázků využívá třída TriVis PHP knihovnu GD. Výsledné obrázky jsou ve formátu PNG. Šířka i výška výsledného obrázku není fixní, pro každý triplex jsou tyto rozměry dynamicky spočítány podle délky triplexu, velikosti smyčky a velikostí inzercí.

Každé vlákno má kvůli přehlednosti označeno svůj 3′ a 5′ konec jinou barvou. Watsonova-Crickova vazba mezi komplementárními vlákny je naznačena nepřerušovanou čarou, Hoogstenova vazba mezi třetím vláknem a jedním z vláken dvojšroubovice je naznačena přerušovanou čarou. Došlo-li k neshodě v párování nukleotidů, Hoogsteenova vazba je vynechána. Inzerce vystupují mimo tělo triplexu a mají šedou barvu, díky které ustupují mírně do pozadí. Smyčka triplexu je tvořena kruhovým obloukem obarveným modře. Popsané vlastnosti zobrazuje následující obrázek 5.6.



Obrázek 5.6: *Vykreslený triplex tTt(g)tt-gcg-tttGt s paralelním prvním vláknem, krátkou smyčkou a inzercí na hlavním vlákně; — Watsonova-Crickova vazba, – – Hoogsteenova vazba.*

Třída TriVis žádným způsobem nekontroluje, jestli jí byl předán opravdový triplex. Jediné, co ji zajímá, je, jestli předaná sekvence splňuje požadovaný formát pro zpracování. V podstatě jde použít pro vykreslení libovolné sekvence ve správném formátu. Hlavní složku kontroly správného formátu tvoří konečný automat definovaný na obrázku 5.7. Pokud tento automat přijme vstupní sekvenci, musíme ještě ověřit, jestli část před smyčkou je stejně



Obrázek 5.7: *Konečný automat, který kontroluje správný formát vstupní sekvence pro vykreslení. Červené šipky znázorňují přechody pro všechny jinak nepokryté symboly daného stavu.*

dlouhá jako část za smyčkou. Na rozhodnutí tohoto problému konečný automat nestačí, takže musíme po přijetí vstupní sekvence konečným automatem provést ještě tuto kontrolu.

Webová aplikace nabízí samostatný nástroj pro vizualizaci triplexů, na kterém lze také vyzkoušet všechny zde zmíněné rysy třídy TriVis. Rozhraní tohoto nástroje je opatřeno patřičnými příklady, ze kterých by mělo pro případné uživatele být snadné pochopit, jak zobrazit, co potřebují, aniž by museli číst tuto práci.

Kapitola 6

Experimenty

Testování bylo prováděno na serveru bioware a byly použity sekvence dostupné z doplňkových materiálů na http://www.fi.muni.cz/~lexa/triplex/.

6.1 Nárůst počtu nalezených triplexů při dělení sekvence

Při rozdělování sekvence jsou překryvy (viz kapitola 4.2) prohledávány dvakrát. To má za následek nárůst celkového počtu výsledků. Řešení tohoto problému spočívá ve filtraci výsledků, kterou ale provádí až webová aplikace. Zajímavé zjištění z těchto výsledků je, že maximální velikost úseku nemá prakticky vliv na časovou náročnost celého výpočtu.

Maximální velikost	5 MB	2, 5 MB	1, 25 MB	0, 625 MB
Počet výsledků	11	13	17	18
Délka výpočtu	24 s	23 s	23 s	23 s

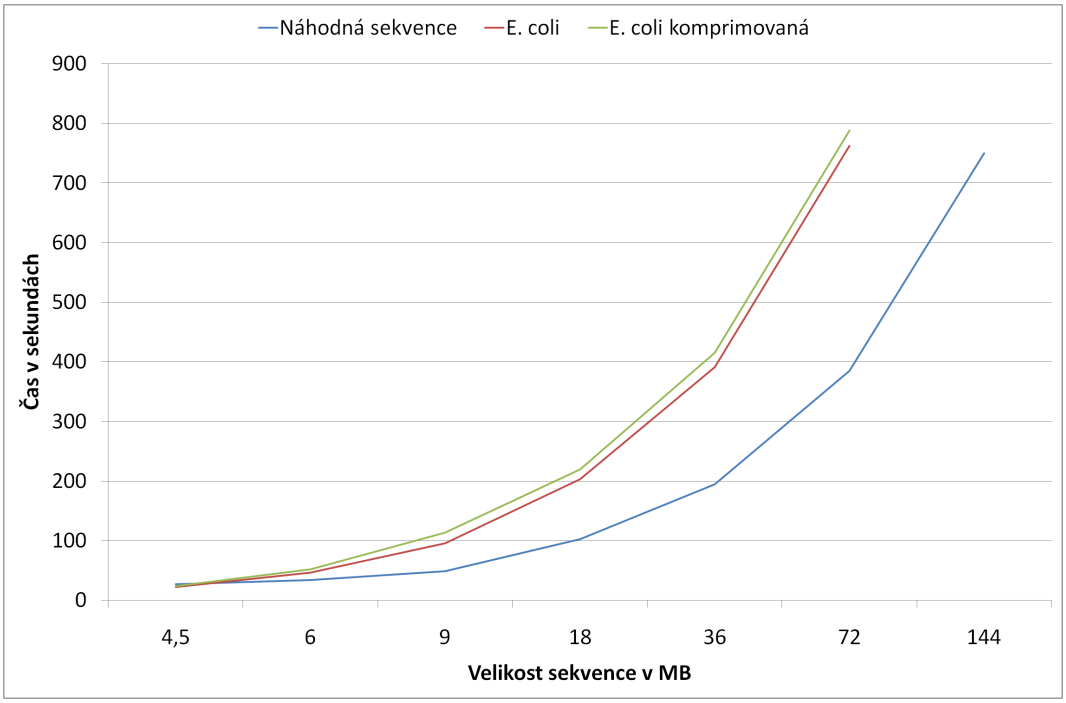
Tabulka 6.1: Nárůstu na genomu E. coli. Použité parametry: -e 12 -i 2 -j 4 -s 20 -l 10

Maximální velikost	5 MB	2, 5 MB	1, 25 MB	0, 625 MB
Počet výsledků	16 209	16 493	16 733	16 967
Délka výpočtu	2 m56 s	2 m57 s	2 m58 s	2 m58 s

Tabulka 6.2: Nárůst na lidském chrom. 22. Použité parametry: -e 20 -i 2 -j 4 -s 20 -l 10

6.2 Časová náročnost

V rámci testování časové náročnosti byl algoritmus otestován na sadě náhodných sekvencí s vyrovnaným poměrem počtu nukleotidů a na reálné sekvenci e. coli, která byla pro tento experiment vytvořena v několika kopiích. Každá kopie obsahovala genom e. coli vícekrát za sebou, tak aby délka sekvence vždy odpovídala náhodné sekvenci. Náhodné sekvence obsahují podle poznatků z kapitoly 3 méně triplexů, a to znamená, že jsou prohledávány rychleji. Zároveň bylo vyhodnoceno zpracování komprimovaných sekvencí v porovnání s nekomprimovanými.



Obrázek 6.1: *Prohledávání náhodných sekvencí trvá kratší dobu, než prohledávání stejně dlouhých sekvencí E .coli. Použití komprimovaných sekvencí přináší jen drobný nárůst v délce zpracování.*

6.3 Nahrazování speciálních FASTA znaků

Pro tyto testy jsem zvolil ukázkovou sekvenci z (Lexa, 2011)[16]:

`TTTCTCCTATCTTCTTCCTCGGG`

Tuto sekvenci jsem pomocí speciálních znaků FASTA formátu upravil. Úprava spočívá v náhradě koncové dvojice `GG` za `RR` (`R` zastupuje puriny), takže algoritmus bude rozhodovat, který z purinů do testované sekvence dosadit. Program spuštěný nad změněnou sekvencí

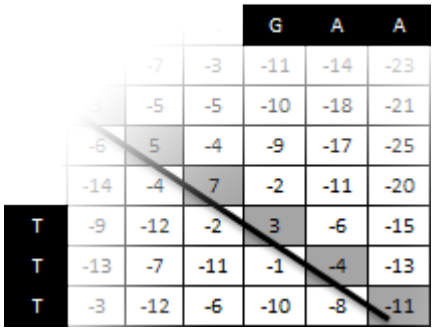
`TTTCTCCTATCTTCTTCCTCGRR`

dává stejný výsledek jako pro originální sekvenci tedy

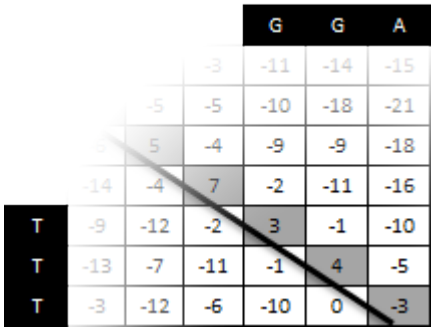
`tttctcct(a)tct--tcttcctcggg`

Na obě místa purinů program dosadil guanin. Matice zpětného průchodu pro tento případ je k vidění na obrázku 4.4. Pro ověření, správnosti tohoto rozhodnutí jsem provedl ručně dosazení všech zbylých možných dvojic na místa purinů tzn. `AA`, `AG`, `GA` a na sekvenci s dosazenými nukleotidy jsem provedl zpětné zarovnání s následujícími výsledky:

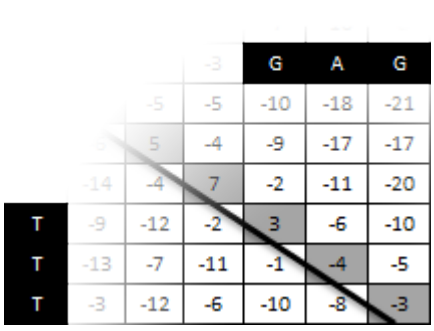
Jak je vidět z obrázku 6.2, program dosadil za dvojici `RR` správně. Varianta `GG` má ze všech možných nejlepší skóre.



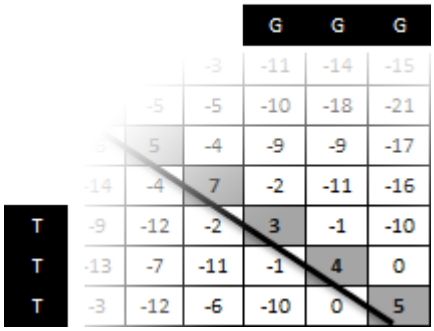
(a) dvojice purinů `RR` nahrazena nukleotidy `AA`.



(b) dvojice purinů `RR` nahrazena nukleotidy `GA`



(c) dvojice purinů `RR` nahrazena nukleotidy `AG`



(d) dvojice purinů `RR` nahrazena nukleotidy `GG`, tak jak navrhl program

Obrázek 6.2: *Na obrázcích je zobrazen pouze pravý dolní roh matice, ten je v tomto problému klíčový.*

Kapitola 7

Závěr

Rozšiřováním a úpravami použitého výchozího algoritmu bylo dosaženo určených cílů. Stejně jako originální aplikace z (Lexa, 2011)[16] je současná verze schopna načítat vstup ve FASTA formátu, vyhledávat i v dlouhých sekvencích DNA a do výpočtu zahrnout změnu izomorfní skupiny. Další úspěšné rozšíření aplikace spočívá v nově zavedené schopnosti zpracovat všechny FASTA znaky pro nukleové kyseliny. Díky tomuto rozšíření je nyní aplikace schopna podle kontextu dosazovat při vyhledávání triplexu za speciální FASTA znaky nejvhodnější nukleotid z množiny, kterou znak reprezentuje. Kromě rozšíření a úprav výchozí aplikace byla vytvořena i její nová verze určená pouze pro krátké sekvence. Cílem tohoto programu je během výpočtu sestavit celou matici dynamického programování a následně v této matici provést zpětný průchod. Zpětný průchod maticí je nutný pro zjištění přesné sekvence triplexu včetně inzercí. Původní aplikace toto nedokáže, protože celou matici nesestavuje (udržuje vždy pouze poslední 2 antidiagonály).

Obě dvě aplikace jsou zkompilovatelné na 64-bitovém systému a díky tomu mohou být použity na serveru bioware ve spolupráci s vytvořenou webovou aplikací. Webová aplikace je dostupná na adrese http://bioware.fit.vutbr.cz. Uživatelům nabízí grafické uživatelské rozhraní pro vyhledávání intramolekulárních triplexů v DNA sekvencích. Vyhledané triplexy umí aplikace vizualizovat díky třídě TriVis, kterou lze použít i samostatně mimo tuto aplikaci pro vlastní vizualizace. Celý web je kompatibilní s hlavními webovými prohlížeči Chrome, Firefox a Internet Explorer 7–9.

Na hlavním programu pro vyhledávání (včetně verze modifikované pro zpětný průchod) vybízí pár věcí ke změnám a vylepšením. Dosazování nukleotidů za speciální znaky FASTA formátu by šlo implementovat důmyslněji. Současný přístup porovnává vždy jen dva sousední symboly a rovnou za ně dosazuje konkrétní nukleotidy. Sofistikovanější přístup by mohl nejprve zjistit, kolik znaků, za které bude muset dosazovat, spolu sousedí a následně rozhodovat o nejvhodnějším dosazení pro celý úsek najednou. Takový přístup by byl výpočetně náročnější, nicméně zároveň s touto prací je jako jiná diplomová práce vyvíjena i hardwarově akcelerovaná verze celého algoritmu, která využívá obecných výpočtů na grafické kartě. V hardwarově akcelerované verzi bychom si i tento výpočetně náročnější postup mohli dovolit.

Rozšíření webové aplikace by mohlo spočívat v důmyslnějším propojení celé aplikace s biologickými databázemi jako je např. UCSC genome browser. Zároveň by se současným vzestupem WebGL bylo možné přepracovat celé zobrazování triplexů tak, aby probíhalo na straně klienta. To by vedlo k snížení zátěže ze serveru.

Literatura

[1] B. Alberts and D. Bray and A. Johnson and J. Lewis and M. Raff and K. Roberts and P. Walter: *Základy buněčné biologie*. Espero, 2005, iSBN 80-902906-2-0.

[2] Bacolla, A.; Wu, F. Y.: Mung bean nuclease cleavage pattern at a polypurine.polypyrimidine sequence upstream from the mouse metallothionein-I gene. *Nucleic Acids Res.*, ročník 19, duben 1991: s. 1639–1647.

[3] Becker, N. A.; Maher, L. J.: Characterization of a polypurine/polypyrimidine sequence upstream of the mouse metallothionein-I gene. *Nucleic Acids Res.*, ročník 26, duben 1998: s. 1951–1958.

[4] Billoud, B.; Kontic, M.; A.Viari: Palingol: a declarative programming language to describe nucleic acids' secondary structures and to scan sequence database. *Nucleic Acids Res.*, ročník 24, duben 1996: s. 1395–1403.
URL http://www.ncbi.nlm.nih.gov/pmc/articles/PMC145829/pdf/241395.pdf

[5] Bissler, J. J.: Triplex DNA and human disease. *Frontiers in Bioscience*, 2007.
URL http://www.bioscience.org/current/vol12.htm

[6] Cer, R. Z.; Bruce, K. H.; Mudunuri, U. S.; aj.: Non-B DB: a database of predicted non-B DNA-forming motifs in mammalian genomes. *Nucleic Acids Res.*, ročník 39, leden 2011: s. D383–391.

[7] Duca, M.; Vekhoff, P.; Oussedik, K.; aj.: The triple helix: 50 years later, the outcome . *Nucleic Acids Research*, ročník 36, 2008: s. 5123–5138.

[8] Frank-Kamenetskii, M. D.; Mirkin, S. M.: Triplex DNA structures. *Annual Review of Biochemistry*, ročník 64, duben 1995: s. 65–95, iSSN 00222836.
URL http://www.annualreviews.org/doi/abs/10.1146/annurev.bi.64.070195.000433

[9] Gaddis, S. S.; Wu, Q.; Thames, H. D.; aj.: A web-based search engine for triplex-forming oligonucleotide target sequences. *Oligonucleotides*, ročník 16, 2006: s. 196–201.

[10] Goni, J. R.: Triplex-forming oligonucleotide target sequences in the human genome. *Nucleic Acids Research*, ročník 32, 2004: s. 354–360, doi:10.1093/nar/gkh188, iSSN 1362-4962.
URL http://www.nar.oupjournals.org/cgi/doi/10.1093/nar/gkh188

[11] Hoyne, P. R.; Edwards, L. M.; Viari, A.; aj.: Searching genomes for sequences with the potential to form intrastrand triple helices. *J. Mol. Biol.*, ročník 302, srpen 2000: s. 797–809.

[12] Hoyne, P. R.; M.Gacy, A.; McMurray, C. T.; aj.: Stabilities of intrastrand pyrimidine motif DNA and RNA triple helices. *Nucleic Acids Res.*, ročník 28, únor 2000: s. 770–775.

[13] Ivo Frébort: Struktura a funkce biomakromolekul. Citováno 19.12. 2011.
URL
http://www.molbio.upol.cz/stranky/vyuka/BPOL/6.%20Struktura%20nukleovych%20kyselin.pdf

[14] Jain, A.; Wang, G.; Vasquez, K. M.: DNA triple helices: biological consequences and therapeutic potential. *Biochimie*, ročník 90, září 2008: s. 1117–1130.

[15] Jenjaroenpun, P.; Kuznetsov, V. A.: TTS mapping: integrative WEB tool for analysis of triplex formation target DNA sequences, G-quadruplets and non-protein coding regulatory DNA elements in the human genome. *BMC Genomics*, ročník 10 Suppl 3, 2009: str. S9.
URL http://www.biomedcentral.com/1471-2164/10/S3/S9

[16] Lexa, M.; Martínek, T.; Burgetová, I.; aj.: A dynamic programming algorithm for identification of triplex-forming sequences. *Bioinformatics*, ročník 27, září 2011: s. 2510–2517.

[17] Li, H.: FASTA/FASTQ Parser in C. 2009, [Online; navštíveno 11. 05. 2012].
URL http://lh3lh3.users.sourceforge.net/parsefastq.shtml

[18] Richard R. Sinden: *Triple-helical nucleic acids*. doi:10.1007/BF02724041.
URL http://www.springerlink.com/index/10.1007/BF02724041

[19] Schroth, G. P.; Ho, P. S.: Occurrence of potential cruciform and H-DNA forming sequences in genomic DNA. *Nucleic Acids Research*, ročník 23, 1995: s. 1977–1983, doi:10.1093/nar/23.11.1977, iSSN 0305-1048.
URL http://nar.oxfordjournals.org/cgi/doi/10.1093/nar/23.11.1977

[20] Stanislav Rosypal: *Úvod do molekulární biologie*. Stanislav Rosypal, 2000, iSBN 80-902562-2-8.

[21] Wikipedie: Chemická afinita — Wikipedie: Otevřená encyklopedie. 2012, [Online; navštíveno 11. 05. 2012].
URL http://cs.wikipedia.org/w/index.php?title=Chemick%C3%A1_afinita&oldid=8046274

Příloha A

Obsah CD

./dokument/pdf	textová verze práce vysázená do pdf
./dokument/tex	textová verze práce pro sazbu
./tests	automatické testy použité v kapitole experimenty
./triplexy_detekce/bin	aplikace pro hledání triplexů přeložená na serveru bioware
./triplexy_detekce/src	zdrojové kódy aplikace pro hledání triplexů
./triplexy_detekce/doc	programová dokumentace aplikace pro hledání triplexů
./vizualizace	třída pro vizualizaci triplexů
./webova_aplikace/aplikace	webová aplikace (dostupná na bioware.fit.vutbr.cz)
./webova_aplikace/schema_db	schéma použité databáze exportované do sql
./zpetne_zarovnani/bin	aplikace pro zpětný průchod přeložená na serveru bioware
./zpetne_zarovnani/src	zdrojové kódy aplikace pro zpětný průchod
./zpetne_zarovnani/doc	programová dkoumentace aplikace pro zpětný průchod