

UNIVERZITA KOMENSKÉHO, BRATISLAVA
FAKULTA MATEMATIKY, FYZIKY A INFORMATIKY

ZAROVNÁVANIE SEKVENCIÍ S POUŽITÍM METÓD
KLASIFIKÁCIE

Diplomová práca

2014

Bc. Michal Hozza

UNIVERZITA KOMENSKÉHO, BRATISLAVA
FAKULTA MATEMATIKY, FYZIKY A INFORMATIKY

ZAROVNÁVANIE SEKVENCIÍ S POUŽITÍM METÓD KLASIFIKÁCIE

Diplomová práca

Študijný program: Informatika
Študijný odbor: 2508 Informatika
Školiace pracovisko: Katedra Informatiky
Školiteľ: Mgr. Tomáš Vinař, PhD.

Bratislava, 2014

Bc. Michal Hozza

Podakovanie...

Bc. Michal Hozza

Abstrakt

Slovenský abstrakt

Kľúčové slová: ...

Abstract

English abstract

Key words: ...

Obsah

Úvod	1
1 Zarovnávanie sekvencií	2
1.1 Podobnosť sekvencií a sekvenčná homológia	2
1.2 Párové zarovnávanie	2
1.3 Typy zarovnaní	3
1.4 Skórovacie systémy	3
1.4.1 Skórovacie matice	3
1.4.2 Afínne skórovanie medzier	3
1.5 Algoritmy na hľadanie zarovnaní	4
1.5.1 Algoritmus pre globálne zarovnanie: Needleman-Wunch	4
1.5.2 Algoritmus pre lokálne zarovnanie: Smith-Waterman	5
1.6 Štatistická významnosť zarovnania	6
2 Súvisiaca práca	8
2.1 Support Vector Training of Protein Alignment Models	8
2.2 CONTRAlign: Discriminative Training for Protein Sequence Alignment	8
3 Doplnkové informácie k sekvenciám	9
4 Náhodný les (Random forest)	10
5 Návrh riešenia	11
6 Implementácia	12
6.1 Simulátor	12
6.1.1 Algoritmus	12
6.1.2 Generovanie informácie o génoch	12
6.1.3 Simulácia mutácie	13
6.1.4 Simulácia delécie	13

<i>OBSAH</i>	vii
7 Zhodnotenie úspešnosti	14
Záver	15
Literatúra	16

Zoznam obrázkov

1.1	Globálne zarovnanie	4
1.2	Lokálne zarovnanie	6
1.3	P-hodnota lokálneho zarovnania so skóre $s = 25$ medzi 2 sekvenciami dĺžky $n = 100$ alebo $n = 200$	7
6.1	Markovova reťaz použitá na generovanie informácie o génoch	13

Zoznam tabuliek

6.1	Pravdepodobnosti mutácie	13
-----	------------------------------------	----

Úvod

Najnovšie technológie sekvenovania DNA produkujú stále väčšie množstvo DNA sekvencií rôznych organizmov. Spolu s tým stúpa aj potreba rozumieť týmto dátam. Dôležitým krokom k ich porozumeniu je *zarovnávanie sekvencií*. Pomocou zarovňavania sekvencií dokážeme zistiť ich štruktúru a následne funkciu jednotlivých častí.

Existujú rôzne algoritmy na zarovnávanie sekvencií. Väčšina z nich je založená na nejakom pravdepodobnostnom modeli, pričom sa snažia nájsť zarovnanie s čo najväčšou pravdepodobnosťou. Či už ide o algoritmy založené na dynamickom programovaní, ktoré s určitosťou nájdu najpravdepodobnejšie zarovnanie, ale za cenu kvadratického času v závislosti od dĺžok sekvencií, alebo heuristické algoritmy, ktoré nie vždy nájdu najpravdepodobnejšie zarovnanie, ale pracujú oveľa rýchlejšie.

My sa budeme zaoberať algoritmom založeným na dynamickom programovaní, kde kvalita výsledného zarovnania je ovplyvnená len pravdepodobnostným modelom.

Základný model berie do úvahy len jednotlivé *bázy* a pravdepodobnosti *substitúcie* (*mutácie*), *inzercie* a *delécie*. Naš model bude navyše uvažovať aj dodatočné informácie získané napríklad z rôznych anotátorov.

Keďže množstvo dodatočnej informácie môže byť veľmi veľké, nie je možné skonštruovať vhodnú skórovaciu maticu pre zarovňavací algoritmus. Namiesto nej teda použijeme klasifikátor, ktorý natrénujeme na sekvenciách so známim zarovnaním a potom použijeme na zarovnanie nových sekvencií. V našom prípade sme si vybrali klasifikátor *Náhodný les* (*Random forest*).

V práci sa budeme zaoberať rôznymi spôsobmi trénovania, pričom navrneme vlastné modely a porovnáme už s existujúcimi. Ďalej sa budeme zaoberať spôsobom zberu dodatočných informácií, využitiu natrénovaného klasifikátora na zarovnávanie nových sekvencií aj pri *prezarovnaní* už existujúcich zarovnaní a implementujeme univerzálny framework, ktorý umožní využívať náš klasifikátor pri rôznych zarovnaníach.

Kapitola 1

Zarovnávanie sekvencií

V tejto kapitole si stručne popíšeme čo je to globálne a lokálne zarovnanie a ukážeme základné algoritmy na hľadanie globálneho a lokálneho zarovnania. Tieto algoritmy budeme neskôr s miernou modifikáciou používať pri našom riešení.

1.1 Podobnosť sekvencií a sekvenčná homológia

V prírode vznikajú nové sekvencie modifikáciou už existujúcich. Preto môžeme často spozorovať podobnosť medzi neznámou sekvenciou a sekvenciou o ktorej už niečo vieme. Ak zistíme podobnosti medzi sekvenciami, môžeme preniesť informácie o štruktúre a/alebo funkcii na novú sekvenciu.

Hovoríme, že dve súvisiace sekvencie sú *homologické*, a že prenášame informácie *podľa homológie*.

Na prvý pohľad rozhodnutie podobnosti dvoch biologických sekvencií nie je nič iné, ako rozhodnutie podobnosti dvoch textových reťazcov. Mnoho metód analýzy biologických sekvencií je preto zakorenená v informatike, kde je už mnoho literatúry týkajúcej sa tejto problematiky.

Vývoj sekvencií hromadí *inzercie*, *delécie* a *substitúcie*, takže predtým ako môže byť vyhodnotená podobnosť, treba urobiť zarovnanie sekvencií. Preto je zarovnanie sekvencií veľmi dôležité.

1.2 Párové zarovnávanie

Párové zarovnávanie je základná úloha zarovnávania sekvencií, kde sa k sebe zarovnávajú dve sekvencie. V tejto práci sa budeme zaoberať len párovým zarovnávaním.

Kľúčové problémy sú:

1. Aké typy zarovnávaní by sme mali uvažovať

2. Skórovací systém, ktorý použijeme na ohodnotenie zarovnania
3. Algoritmus, ktorý použijeme na hľadanie optimálneho alebo dobrého zarovnania podľa skórovacieho systému
4. Štatistická významnosť zarovnania.

1.3 Typy zarovnaní

Základné typy zarovnaní sú *Globálne zarovnanie* a *Lokálne zarovnanie*.

Definícia 1.3.1 (Globálne zarovnanie). Vstupom sú dve sekvencie $X = x_1x_2 \dots x_n$ a $Y = y_1y_2 \dots y_m$. Výstupom je zarovnanie celých sekvencií X a Y s najvyšším skóre.

Definícia 1.3.2 (Lokálne zarovnanie). Vstupom sú dve sekvencie $X = x_1x_2 \dots x_n$ a $Y = y_1y_2 \dots y_m$. Výstupom je zarovnanie nejakých poriadkov $x_i \dots x_j$ a $y_k \dots y_l$ sekvencií s najvyšším skóre.

1.4 Skórovacie systémy

Takmer všetky metódy zarovnania hľadajú zarovnanie dvoch reťazcov na základe nejakej *skórovacej schémy*. Skórovacie schémy môžu byť veľmi jednoduché, napr. $+1$ za *zhodu* a -1 za *nezhodu*. Hoci ak chceme mať schému, kde biologicky najkorektnjšie zarovnanie má najvyššie skóre, musíme vziať do úvahy, že biologické sekvencie majú evolučnú históriu, 3D štruktúru a mnohé ďalšie vlastnosti obmedzujúce ich primárnu evolúciu. Preto skórovací systém vyžaduje starostlivé premyslenie a môže byť veľmi zložitý.

1.4.1 Skórovacie matice

Niekedy chceme rôzne zhody a nezhody skórovať rôzne - nie len všetky zhody $+1$ a nezhody -1 . Skóre môže závisieť od toho aké bázy sú v danom stĺpci zarovnania. Na to môžeme použiť *skórovaciu maticu*, kde máme definované skóre pre každú dvojicu. Skórovacie matice sa využívajú najmä pri zarovnávaní proteínov, kde niektoré dvojice majú podobné chemické vlastnosti.

1.4.2 Afínne skórovanie medzier

V jednoduchom skórovaní sme dávali za pomčľku vždy rovnaké skóre (-1). Pri evolúcii sa však môže stať, že sa naraz zmaže niekoľko susedných báz. Pri *afínnom skórovaní*

medzier teda zavedieme dva typy skóre. Skóre za *začatie medzery* a skóre za *rozšírenie medzery*.

1.5 Algoritmy na hľadanie zarovnaní

Máme daný skórovací systém, potrebujeme algoritmus, ktorý nájde optimálne zarovnanie dvoch sekvencií. Budeme uvažovať zarovnávanie s medzerami. To znamená, že môžeme do sekvencie pridať ľubovoľne veľa medzier, aby sme dosiahli lepšie skóre. Pre 2 sekvencie dĺžky n existuje

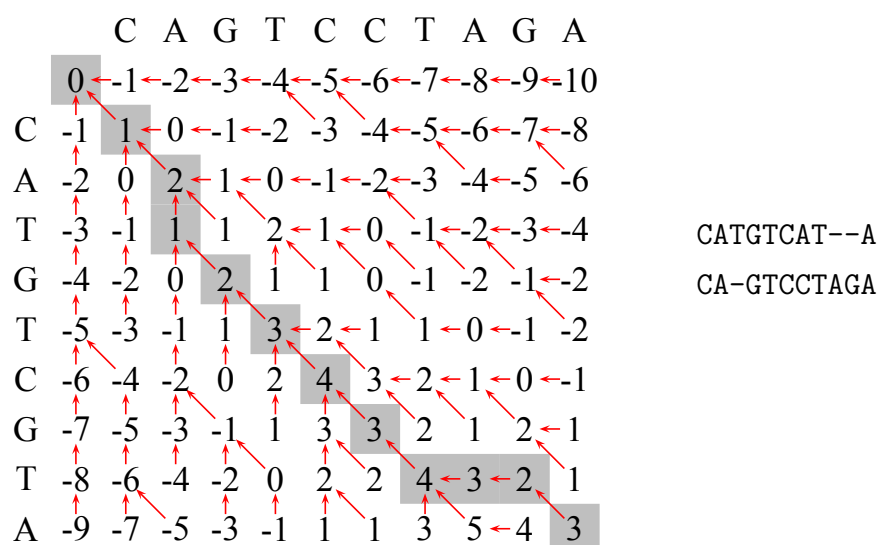
$$\binom{2n}{n} = \frac{(2n)!}{(n!)^2} \simeq \frac{2^{2n}}{\sqrt{\pi n}}$$

možných globálnych zarovnaní. Čiže nie je možné v rozumnom čase nimi prejsť.

Algoritmy na hľadanie zarovnaní využívajú *dynamické programovanie*. Algoritmy s dynamickým programovaním garantujú nájdenie optimálneho zarovnania. Existujú aj heuristické algoritmy, ktoré môžu byť veľmi rýchle, avšak majú určité predpoklady a môže sa stať, že nenájdu najlepšie zarovnanie pre niektoré páry sekvencií. My sa budeme zaoberať len algoritmami využívajúcimi dynamické programovanie. Pre rôzne typy zarovnaní máme rôzne algoritmy zarovnávaní.

1.5.1 Algoritmus pre globálne zarovnanie: Needleman-Wunch

Máme dané 2 sekvencie $X = x_1x_2 \dots x_n$ a $Y = y_1y_2 \dots y_m$, budeme zarovnávať všetky znaky sekvencie X a všetky znaky sekvencie Y . Budeme používať jednoduché skórovanie: +1 za zhodu, -1 za nezhdodu alebo pomlčku.



Obr. 1.1: Globálne zarovnanie

Algoritmus postupne vyplňa 2-rozmernú tabuľku A . Riadky zodpovedajú bázam sekvencie X a stĺpce bázam Y . Na políčku $A[i, j]$ bude skóre najlepšieho zarovnania prvých i báz sekvencie X a prvých j báz Y .

Ak zarovnáваме sekvenciu s prázdnu sekvenciou, tak skóre bude $-n$, kde n je dĺžka sekvencie. Bude tam n pomlčiek, každá nám dá skóre -1 . Takto vyplníme riadky a stĺpce $A[i, 0]$ a $A[0, j]$.

Ak chceme vyplniť políčko $A[i, j]$, musíme si uvedomiť ako môže vyzerat posledný stĺpec zarovnania $x_1x_2 \dots x_i$ a $y_1y_2 \dots y_j$. Máme iba 3 možnosti ako môže vyzerat posledný stĺpec najlepšieho zarovnania. Buď obsahuje x_i alebo y_j alebo oboje. Ak posledný stĺpec obsahuje oboje, cena tohto stĺpca je buď $+1$ ak $x_i = y_j$ alebo -1 ináč. Ak by sme posledný stĺpec zmazali, dostali by sme zarovnanie $x_1x_2 \dots x_{i-1}$ a $y_1y_2 \dots y_{j-1}$, pričom musí íť o najlepšie zarovnanie. To už máme vypočítané v políčku $A[i-1, j-1]$, čiže výsledné skóre bude $A[i-1, j-1] + s(x_i, y_j)$.

Ak posledný stĺpec obsahuje len x_i zarovnané s pomlčkou, skóre stĺpca bude -1 a po zmazaní dostávame zarovnanie $x_1x_2 \dots x_{i-1}$ a $y_1y_2 \dots y_j$, výsledné skóre bude teda $A[i-1, j] - 1$. V prípade, že posledný stĺpec obsahuje len y_j , tak skóre vypočítame analogicky.

Najlepšie skóre bude maximum skóre pre všetky 3 prípady. Dostávame teda nasledujúci vzťah pre výpočet $A[i, j]$:

$$A[i, j] = \max \begin{cases} A[i-1, j-1] + s(x_i, y_j) \\ A[i-1, j] - 1 \\ A[i, j-1] - 1 \end{cases}$$

Tabuľku vieme vyplňať po riadkoch, pričom každé políčko vieme vypočítať z troch políčok, ktoré už máme vypočítané.

Ak nás zaujíma aj zarovnanie – nie len jeho skóre – vieme si pre každé políčko zapamätať ktorá z 3 možností dosiahla maximálnu hodnotu. Na základe tejto informácie potom vieme zrekonštruovať zarovnanie tak, že postupne z posledného políčka ($A[n, m]$) budeme prechádzať na políčko, z ktorého sme vypočítali aktuálnu hodnotu.

Časová zložitosť je $O(nm)$, pretože vyplníme nm políčok, každé v konštantnom čase. Zjavne aj pamäťová zložitosť je $O(nm)$.

Pamäťová zložitosť sa dá zredukovať na $O(n + m)$ za cenu zhruba dvojnásobného času výpočtu [Hir75].

1.5.2 Algoritmus pre lokálne zarovnanie: Smith-Waterman

Algoritmus pre lokálne zarovnania sa líši len v niekoľkých malých detailoch. Opäť vyplníme tabuľku A , s tým, že v $A[i, j]$ bude najvyššie skóre lokálneho zarovnania

		C	A	G	T	C	C	T	A	G	A
		0	0	0	0	0	0	0	0	0	0
A		0	0	1	0	0	0	0	1	0	1
T		0	0	0	0	1	0	0	1	0	0
G		0	0	0	1	0	0	0	0	1	0
T		0	0	0	0	2	1	0	1	0	0
C		0	1	0	0	1	3	2	1	0	0
T		0	0	0	0	1	2	2	3	2	1
A		0	0	1	0	0	1	1	2	4	3
T		0	0	0	0	1	0	0	2	3	2

GT-CTA

GTCCTA

Obr. 1.2: Lokálne zarovnanie

medzi sekvenciami $x_1x_2 \dots x_i$ a $y_1y_2 \dots y_j$, ktoré buď obsahuje bázy x_i aj y_j , alebo je prázdne. Teda na ľubovoľnom mieste uvažujeme aj prázdne zarovnanie so skóre 0 (v matici nebudú záporné čísla). Vzťah pre výpočet $A[i, j]$ vyzerá takto:

$$A[i, j] = \max \begin{cases} 0 \\ A[i-1, j-1] + s(x_i, y_j) \\ A[i-1, j] - 1 \\ A[i, j-1] - 1 \end{cases}$$

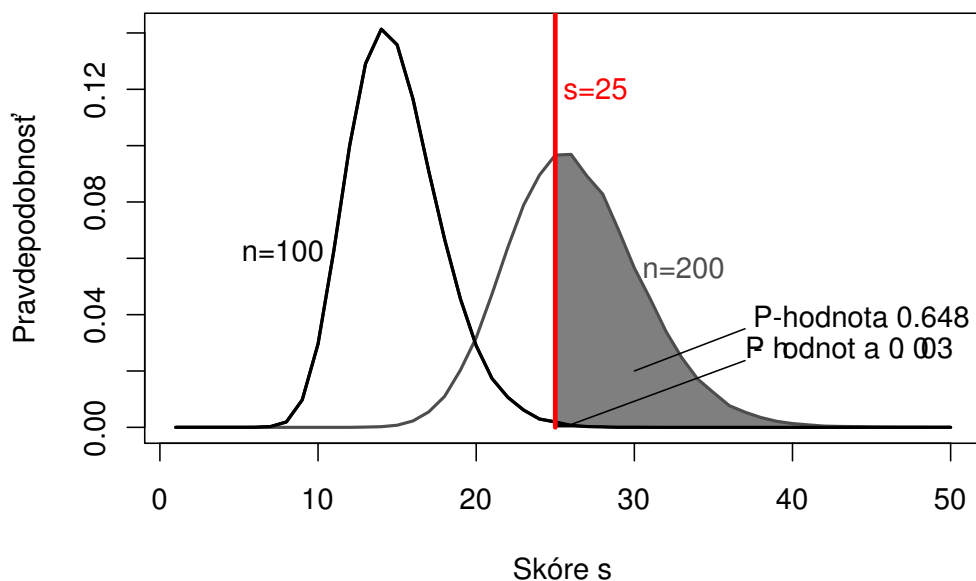
Časová aj pamäťová zložitosť sú, rovnako ako pri globálnom zarovnaní $O(nm)$.

1.6 Štatistická významnosť zarovnania

Smith-Watermanov algoritmus nájde najlepšie lokálne zarovnanie, pre ľubovoľné dve sekvencie. Treba však rozhodnúť, či je zarovnanie dostatočne vierohodné na to, aby predstavovalo skutočnú podobnosť sekvencií a nie len najlepšie zarovnanie dvoch nesúvisiacich sekvencií. Ako vodítko pri rozhodnutí sa používajú identifikátory *štatistickej významnosti* zarovnania: *P-hodnota* (*P-value*) alebo *E-hodnota* (*E-value*).

P-hodnota zarovnania je pravdepodobnosť, že medzi náhodne generovanými sekvenciami tej istej dĺžky by sme našli zarovnanie s rovnakým skóre alebo vyšším. Keďže P-hodnota závisí od dĺžok sekvencií a skóre, musíme ju počítať pri každom zarovnaní. Je však časovo náročné robiť to generovaním veľkého množstva zarovnaní, preto sa používajú matematicky odvodené vzorce na odhad tejto hodnoty. ([KA90], [MB06]).

E-hodnota vyjadruje priemerný počet zarovnaní so skóre aspoň takým ako má naše zarovnanie medzi náhodne generovanými sekvenciami. E-hodnota teda môže byť aj väčšia ako jedna. Ak je E-hodnota väčšia ako jedna, tak čisto náhodou by sme očakávali



Obr. 1.3: P-hodnota lokálneho zarovnania so skóre $s = 25$ medzi 2 sekvenciami dĺžky $n = 100$ alebo $n = 200$

aspoň 1 také silné zarovnanie a teda zarovnania s tak vysokou E-hodnotou nebudeme považovať za štatisticky významné.

V štatistike sa v rôznych testoch štandardne používajú prahy na P-hodnotu 0.05 alebo 0.01. Pri zarovnávaní sekvencií však často používame ešte nižší prah, teda uvažujeme len zarovnania s P-hodnotou menšou ako napr. 10^{-5} . Pri malých hodnotách sú P-hodnota a E-hodnota približne rovnaké, teda taký istý prah môžeme použiť aj na E-hodnotu.

Kapitola 2

Súvisiaca práca

V tejto kapitole si uvedieme stručný prehľad modelov, ktoré zahŕňajú doplnkové informácie do zarovnania pomocou metód klasifikácie a stručne uvedieme v čom sa bude náš model líšiť.

2.1 Support Vector Training of Protein Alignment Models

2.2 CONTRAlign: Discriminative Training for Protein Sequence Alignment

Kapitola 3

Doplnkové informácie k sekvenciám

Kapitola 4

Náhodný les (Random forest)

V tejto kapitole si popíšeme ako funguje klasifikátor náhodný les, v čom spočívajú jeho výhody a prečo sme si ho vybrali.

Kapitola 5

Návrh riešenia

Kapitola 6

Implementácia

6.1 Simulátor

Simulátor slúži na overenie funkčnosti klasifikátora. Náhodne vygeneruje 2 sekvencie, ktoré vznikli zo spoločného predka a vyrobí korektné zarovnanie. Okrem toho vyrobí aj nejaké dodatočné informácie ktoré majú pomôcť pri zarovnávaní.

6.1.1 Algoritmus

Simulátor vygeneruje informáciu o tom, ktoré časti sekvencie prislúchajú génom a ktoré nie. Informácia má podobu boolovského vektora. Simulátor najskôr vygeneruje *základnú (master) postupnosť* a z nej odvodí dve ďalšie postupnosti, ktoré zodpovedajú sekvenciám.

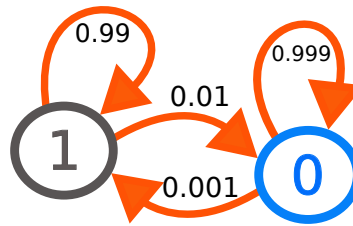
Okrem toho simulátor vygeneruje dve sekvencie, pričom prvú vyrobí náhodne a druhú odvodí z nej pomocou mutácie a inzercie/delécie. V našom prípade sme inzerciu vynechali a simulujeme ju ako deléciu v druhej sekvencii. Pri odvodzovaní bude používať aj informáciu o tom, ktorá časť je gén a ktorá nie.

Keďže simulátor vie spôsob akým generoval druhú sekvenciu z prvej, vie aj korektné zarovnanie.

Simulátor má vopred daných niekoľko konštánt – pravdepodobnosti udalostí, ktoré môžu nastať.

6.1.2 Generovanie informácie o génoch

Ak sa na danom mieste nachádza gén, označíme to 1 inak 0. Gény bývajú súvislé úseky, takže ich treba generovať tak, že niekedy začneme gén, potom generujeme 1, potom skončíme gén a generujeme 0. Potom môžeme opäť začať gén atď.



Obr. 6.1: Markovova reťaz použitá na generovanie informácie o génoch

Generovanie robíme pomocou *Markovovej reťaze* (*Markov Chain*) obr. 6.1. Generujeme podľa aktuálneho stavu a v každom kroku sa podľa pravdepodobnosti rozhodneme či sa prepne do iného stavu alebo ostaneme v tom istom. Rozhodnutie robíme pomocou *falošnej mince* (*biased coin*), kde hlava padne s určitou pravdepodobnosťou, ktorú vopred nastavíme.

Máme vygenerovanú master postupnosť a z nej teraz vyrobíme dve postupnosti pre sekvencie tak, že skopírujeme master sekvenciu, pričom každú 1 s určitou pravdepodobnosťou (v našom prípade 0, 1) zmeníme na 0.

6.1.3 Simulácia mutácie

Máme vygenerovanú sekvenciu a ideme vyrobiť zmutovanú sekvenciu. Spravíme to tak, že s určitou pravdepodobnosťou sa nahradí báza z pôvodnej sekvencie inou bázou. Pravdepodobnosť závisí aj od toho, či je na danej pozícii gén v oboch sekvenciách, v jednej, alebo v žiadnej. Na rozhodnutie používame jednu z troch falošných mincí podľa toho, ktorá z možností nastala (tabuľka 6.1).

Gén A	0	0	1	1
Gén B	0	1	0	1
Pravdepodobnosť	0,35	0,3	0,3	0,2

Tabuľka 6.1: Pravdepodobnosti mutácie

6.1.4 Simulácia delécie

Deléciu simulujeme opäť pomocou Markovovej reťaze, pretože pri evolúcii majú tendenciu vypadávať súvislé úseky. Pravdepodobnosť, že začneme mazať je 0,01 a že prestaneme 0,1. Ak mažeme, nahradzujeme danú bázu znakom '-'.

Kapitola 7

Zhodnotenie úspešnosti

Záver

Literatúra

- [BPSS11] Alvis Brazma, Helen Parkinson, Thomas Schlitt, and Mohammadreza Shojatalab. EBI Research - Functional Genomics - Introduction To Biology. 2011. [Online; accessed 26-Oct-2012].
- [BV11] Broňa Brejová and Tomáš Vinař. *Metódy v bioinformatike [Methods in Bioinformatics]*. Knižničné a edičné centrum FMFI UK, 2011. Lecture notes.
- [DEKM98] R. Durbin, S.R. Eddy, A. Krogh, and G. Mitchison. *Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids*. Cambridge University Press, 1998.
- [DGB06] Chuong Do, Samuel Gross, and Serafim Batzoglou. Contralign: Discriminative training for protein sequence alignment. In Alberto Apostolico, Concettina Guerra, Sorin Istrail, Pavel Pevzner, and Michael Waterman, editors, *Research in Computational Molecular Biology*, volume 3909 of *Lecture Notes in Computer Science*, pages 160–174. Springer Berlin / Heidelberg, 2006. 10.1007/11732990_15.
- [Hir75] D. S. Hirschberg. A linear space algorithm for computing maximal common subsequences. *Commun. ACM*, 18(6):341–343, June 1975.
- [KA90] S Karlin and S F Altschul. Methods for assessing the statistical significance of molecular sequence features by using general scoring schemes. *Proceedings of the National Academy of Sciences*, 87(6):2264–2268, 1990.
- [MB06] Alexander Yu. Mitrophanov and Mark Borodovsky. Statistical significance in biological sequence analysis. *Briefings in Bioinformatics*, pages 2–24, 2006.
- [NW70] Saul B. Needleman and Christian D. Wunsch. A general method applicable to the search for similarities in the amino acid sequence of two proteins. *Journal of Molecular Biology*, 48(3):443 – 453, 1970.

- [SW81] T.F. Smith and M.S. Waterman. Identification of common molecular subsequences. *Journal of Molecular Biology*, 147(1):195 – 197, 1981.
- [YJEP07] Chun-Nam Yu, Thorsten Joachims, Ron Elber, and Jaroslaw Pillardy. Support vector training of protein alignment models. In Terry Speed and Haiyan Huang, editors, *Research in Computational Molecular Biology*, volume 4453 of *Lecture Notes in Computer Science*, pages 253–267. Springer Berlin / Heidelberg, 2007. 10.1007/978-3-540-71681-5_18.