TECHNICS FOR DATAMINING IN GENOMICS

Petr Jaša
Master Degree Programme (2), FIT BUT
E-mail: xjasap02@stud.fit.vutbr.cz

Supervised by: Ivana Rudolfová
E-mail: rudolfa@fit.vutbr.cz

ABSTRACT

The paper introduces heuristic methods for datamining in genomics. It is mainly focused on pairwise alignment of DNA sequences. In that case, algorithms like Fasta and Blast are used for alignment of one query sequence of DNA between many database sequences. This alignment determines matching rate of query sequence with other database sequences. Based on the result, it is possible to determine many features of query sequence with given probability.

1. ÚVOD

Spojení informatiky a biologie dalo vznik novému vědnímu oboru - bioinformatice. Jednou z oblastí biologie, kterou se bioinformatika zabývá je obor genomika. Genomika je rychle se rozvíjející vědní obor, jehož snahou je rozluštit kompletní genom živých organizmů a vytěžit z této znalosti maximum biologicky relevantních informací. Pojem genom souhrnně vyjadřuje veškerou dědičnou informaci organismů, která je ukryta v pořadí nukleotidů (označované písmeny A, C, G, T, U) nukleových kyselin jako je například molekula DNA [1]. Genom určuje primární rysy organismů a každá jeho část určující nějaký rys se označuje jako gen.

Rozluštění genomu spočívá ve stanovení přesného pořadí nukleotidů molekuly DNA případně dalších nukleových kyselin. Znalost pořadí nukleotidů má obrovský význam pro přesné porovnávání různých genomů nukleotid po nukleotidu. Je tak možné lépe vysledovat evoluci a vzájemné vztahy jednotlivých druhů organismů. Studium genů jednodušších organismů výrazně napomáhá pochopit význam genů složitějších organizmů s podobným (ale složitějším) genomem. Porovnání genomů může také například odhalit úsek DNA, který je příčinou dědičné choroby, a podobně.

V poslední době dochází k dramatickému nárůstu genomických dat. Samotný lidský genom obsahuje přibližně 3,2 miliardy nukleotidů. Je proto nutné pro porovnání sekvencí takovýchto délek využívat vhodné algoritmy, které jsou schopné se s takovým množstvím dat vypořádat v rozumném čase. Tento článek se zaměřuje na algoritmus BLAST [2], který pro porovnávání využívá heuristických postupů. Základ tohoto algoritmu bude popsán v třetí kapitole. Před tím budou v druhé kapitole vysvětleny základní problémy, které se pří porovnávání sekvencí mohou objevit. V současnosti existuje několik verzí tohoto algoritmu. Verze, které umožňují řešit veškeré problémy porovnávání, však nemají veřejné zdrojové kódy. Cílem této práce je proto vytvořit vlastní implementaci algoritmu BLAST s možností řešit veškeré problémy porovnávání sekvencí DNA.

2. POROVNÁVÁNÍ SEKVENCÍ

Porovnání dvou sekvencí je jednoduše porovnání shody mezi jednotlivými znaky každé sekvence. Aby toto porovnání mělo reálnou vypovídající hodnotu, musí být sekvence vůči sobě optimálně zarovnány. V průběhu evoluce může v sekvenci nukleotidů každého druhu docházet k třem různým změnám - mutace (záměna nukleotidu), inserce (vložení jednoho nebo více nových nukleotidů), delece (odebrání jednoho nebo více nových nukleotidů).

Protože k inserci a deleci neexistuje žádná homologie, zavádí se pojem mezera, která při porovnávání sekvencí reflektuje tyto změny. Uvažujme dvě sekvence: `AATCTATA` a `AAGATA`. Pokud předpokládáme, že v sekvencích nedošlo k inserci ani deleci, je možné tyto sekvence vůči sobě zarovnat právě třemi způsoby. Pro stanovení optimálního zarovnání se zavádí tzv. skórovací funkce. V případě bez insercí a delecí je skórovací systém realizován funkcí, která určuje součet jednotlivých shod a neshod porovnávaných řetězců. Pokud vezmeme v úvahu případ, že v sekvenci nastaly inserce nebo delece, porovnání se výrazně zkomplikuje a počet možných zarovnání vzroste na hodnotu 28 (příklad 1).

```
AATCTATA        AATCTATA        AATCTATA
AAG-AT-A        AA-G-ATA        AA--GATA
```

Příklad 1: tři z 28 možných způsobů zarovnání sekvencí.

Při ohodnocování zarovnání sekvencí, které mohou obsahovat mezery, je nutné do skórovací funkce přidat hodnotu penalizace za zarovnání znaku jedné sekvence s mezerou druhé sekvence. Funkce, kde n je délka delší sekvence, pak vypadá následovně [1]:

$$\sum_{i=1}^{n} \begin{cases} \text{Penalizace za mezeru; jestliže } seq1_i = \text{‘–‘ nebo } seq2_i = \text{‘–‘} \\ \text{Skóre shody; jestliže tu není mezera a } seq1_i = seq2_i \\ \text{Skóre neshody; jestliže tu není mezera a } seq1_i \neq seq2_i \end{cases}$$

Protože některé substituce znaků se vyskytují v přírodě častěji než jiné, musí hodnoty penalizace odpovídat pravděpodobnosti změn objevující se v evoluci. Dále je důležité vzít v úvahu případ, kdy sekvence obsahují pouze krátkou (vzhledem k jejich celkové délce) společnou podsekvenci. V případě takovýchto lokálních podobností způsobí okolní neshody, že zarovnání sekvence nemá dobré skóre, přestože může mít výrazný biologický význam. Porovnávací algoritmus tedy musí být připraven na možnost globálního nebo lokálního porovnání s mezerami nebo bez mezer. Zároveň by měl dokázat adekvátně ohodnotit změny vzniklé evolucí podle jejich pravděpodobnosti výskytu a biologického významu.

Zásadním problémem porovnávání dvou sekvencí je počet možností jejich vzájemného zarovnání, který s rostoucí délkou sekvencí roste ohromným tempem. Při porovnávání sekvencí o délce 100 a 95 nukleotidů vzniká přibližně 55 milionů možností zarovnání. Tento problém je možné řešit tzv. dynamickým programováním [1]. Tyto metody dovolují dosáhnout optimálního porovnání dvou sekvencí v takovém čase, který je úměrný délkám těchto porovnávaných sekvencí. To je relativně dobrý výsledek. Ovšem v případě, kdy se tyto metody aplikují na celou databázi sekvencí, je lineární růst časové náročnosti stále nevhodný. Pro dnešní rozsáhlé databáze je proto nutné využít různé heuristiky, umožňující v rozumném čase porovnat jednu sekvenci proti mnoha jiným sekvencím.

3. BLAST (BASIC LOCAL ALIGNMENT SEARCH TOOL)

Jeden z představitelů algoritmů pro porovnávání sekvencí založeného na heuristice se nazývá BLAST [2]. Jeho princip spočívá ve vyřazení „nevhodných“ sekvencí hned na

začátku porovnávání, kdy se algoritmus snaží co nejrychleji lokalizovat podobné úseky sekvence bez mezer mezi sekvencí dotazu a sekvencemi databáze. Algoritmus tak ale riskuje ztrátu určité citlivosti. Základ algoritmu tvoří tři kroky.

3.1. PŘEDZPRACOVÁNÍ DOTAZU

Nechť A = {A, C, G, T, U} je abeceda znaků sekvencí. Nechť D je sekvence databáze nad abecedou A a Q je sekvence dotazu nad abecedou A. V prvním kroku jsou nejdříve vygenerována všechna slova nad abecedou A o stanovené délce w (parametr programu). Potom se postupně každé takové slovo o délce w zarovnává s každým podřetězcem (také o délce w) sekvence Q. Každá pozice sekvence Q je pak asociována se seznamem slov, které při porovnání s podřetězcem sekvence Q získaly skóre větší než stanovený práh T (parametr programu).

3.2. GENEROVÁNÍ ZÁSAHŮ

Po prvním kroku je Q nyní reprezentovaná seznamy slov (tzv. seznamy sousedů). Každá pozice Q je porovnána s každým podřetězcem sekvence D, a jestliže jedno ze sousedních slov na té pozici sekvence Q je identické slovu z D, je zaznamenána shoda tzv. zásah. Tedy pro každé slovo ze seznamu sousedů stanovíme všechny konkrétní zásahy do sekvence D.

3.3. PRODLOUŽENÍ ZÁSAHŮ

Každá shoda (zásah), generovaná v předešlém kroku, je nyní rozšiřována v obou směrech, bez mezer, za účelem stanovit, zdali tato shoda může být částí většího úseku podobnosti. Rozšiřování je zastaveno, jakmile skóre rozšířené shody klesne o více než stanovenou hodnotu X (parametr programu) vůči nejvyšší dosažené hodnotě skóre daného zásahu. Každá rozšířená dvojice, která má skóre stejné nebo lepší než S (parametr programu) je uchována a nazvaná jako HSP (High scoring Segment Pair). HSP lze chápat jako úsek stejné délky dvou sekvencí se skóre, které už nelze zlepšit prodloužením. Sekvence obsahující HSP mohou být označeny za nejvíce podobné sekvence sekvenci Q.

4. ZÁVĚR

Uvedený algoritmus BLAST umožňuje výrazné zrychlení zarovnávání sekvencí DNA. V základní verzi umožňuje pouze lokální zarovnávání bez mezer. Výsledkem tohoto projektu bude platformě nezávislá aplikace, umožňující i globální zarovnání sekvencí s mezerami. Pro implementaci byl zvolen jazyk Java. Jsou využívány volně dostupné knihovny poskytované v rámci projektu biojava [3]. V současné době je aplikace ve stavu, kdy je naimplementována základní funkčnost okenní aplikace, vzhled a metody dynamického programování, které algoritmus BLAST při zarovnávání sekvencí využívá.

LITERATURA

[1] Wang, J. T. L., Zaki, M. J., Toivonen, H. T. T., Shasha, D.: Data Mining in Bioinformatics, Springer-Verlag, 2005, ISBN 1852336714

[2] Frédérique, G.: The Fasta and Blast programs, 18. července 2000, Dostupné z: <https://www.fit.vutbr.cz/study/courses/ZZN/private/prednasky/blast1.pdf> (leden 2007).

[3] BioJava: elektronické stránky věnované projektu BioJava [online], Dostupné z: <http://biojava.org/wiki> (leden 2007).