Prof. Dr. Knut Reinert
Enrico Siragusa
Sascha Meiers
Christoph Hartmann

Institut für Informatik
AG Algorithmische Bioinformatik

Algorithmen und Datenstrukturen in der Bioinformatik
Neuntes Übungsblatt WS 11/12

Abgabe Montag, 02.01.2011, 15:00 Uhr, diesmal auch per E-Mail

Name: Übungsgruppe: A☐ B☐ C☐

Matrikelnummer:

Niveau I

Aufgabe1: Pidgeonhole Principle

Beweisen Sie das Lemma 1 aus dem Skript, Kapitel 6.2.

Lemma 1. *Gegeben sei ein Match zwischen einem Text Occ und einem Pattern P mit k Fehlern, desweiteren sei $P = p^1, \ldots, p^j$ eine beliebige Konkatenation von Infixen (= Teilstrings) von P und seien $a_1, \ldots, a_j$ nicht-negative Ganzzahlen mit $A = \sum_{i=1}^{j} a_i$.*
Dann gilt: es existiert ein $i \in [1 \ldots j]$, so dass Occ ein Infix enthält, welches p^i mit höchstens $\lfloor a_i k / A \rfloor$ Fehlern matcht.

Hinweis: Es empfiehlt sich ein relativ simpler Widerspruchsbeweis. $a_1, \ldots, a_j$ sind dabei willkürliche gewählte Zahlen. Die vorangehende Aussage ("Teilt man das Pattern in $k+1$ Teile ...") ist ein Spezialfall dieses Lemmas, bei dem man $a_i = 1 \quad \forall i$ wählt.

Aufgabe 2: Approximate Matching

Finden Sie das Pattern $P =$ `WIRSING` mit maximal $k = 3$ Abweichungen im Text $T =$ `WIRSINDIRRSINNIG`. Benutzen Sie dafür den Algorithmus mit hierarchischer Verifizierung aus dem Skript, Kapitel 6.4. Wenn Sie wollen, können Sie die exakten Treffer der Teilworte mit dem Aho-Corasick-Algorithmus suchen; ansonsten reicht es aus die Treffer per Hand zu finden.

Niveau II

Aufgabe 3: PEX

Der PEX-Algorithmus ist ein Algorithmus zur approximativen Stringsuche, der das Schubfachprinzip nutzt. Im Skript, Kapitel 6.5 sind die Pseudocodes für die Baumerstellung und Suchphase des PEX-Algorithmus gegeben. Benutzen Sie diese, um

a) einen Suchbaum für das Wort `BRAUN` für $k = 2$, also maximal zwei abweichende Zeichen, zu erstellen

b) mit diesem Suchbaum den Text `BLAUKRAUT` zu durchsuchen.

Programmieraufgabe (Abgabe Montag, 9.01.2012, 15:00)

P-Aufgabe 5: Implement a program to map a set $\mathcal{R}$ of genomic reads to a reference genome G with at most k mismatches. To this end, your program should perform the following steps:

a) Read the reference genome G and the reads $\mathcal{R}$ from their corresponding input files (Template code given)

b) Construct a q-gram Index over G

c) For each read $R \in \mathcal{R}$

i) Partition R into $k + 1$ pieces

ii) Search each piece in G using the q-gram Index (Each occurrence of a piece of R in G is called a *hit*)

iii) Verify that R matches with at most k mismatches the substring of G surrounding each hit

iv) Write end positions of all approximate occurrences of R in G into an output file (Beware of reporting each approximate occurrence only once)

Hints:

- The underlying Alphabet is $\Sigma = \{\mathtt{A}, \mathtt{C}, \mathtt{G}, \mathtt{T}\}$.
- Reads are short sequences of length 36–400. For simplicity, all reads will have the same length.
- A naive algorithm for Hamming Distance can be used to verify that a read R matches a substring of G with at most k mismatches.
- Your program must take the file containing the reference genome G as the first command-line argument, the file containing the set of reads $\mathcal{R}$ as the second argument, the number of mismatches k as the third argument, the output file as the fourth argument.
- The reference genome G is stored on a single line.
- Each read $R \in \mathcal{R}$ is stored on a single and distinct line.
- Each approximate occurrence must be written in a separate row of the output file, following the format: ⟨Read R⟩,⟨End position in G⟩,⟨Mismatches⟩

- You can download the code template at `https://svn.imp.fu-berlin.de/agbio/aldabi/ws11/documents/aufgabe5`.

Example:

- File with reference genome G:

  ```
  AAGATACATTTTTAAAAAAAAAAACAATTT
  ```

- File with reads $\mathcal{R}$:

  ```
  GATACA
  CATTTT
  ```

- Output file:

  ```
  GATACA,7,0
  CATTTT,11,0
  CATTTT,28,1
  ```

Evaluation:

- This exercise will be evaluated with 6 points instead of 3.