technische universität dortmund

Bioinformatik, Fakultät für Informatik

UNIVERSITÄT DUISBURG ESSEN

Offen im Denken

Genominformatik, Medizinische Fakultät

Prof. Dr. Sven Rahmann

Wintersemester 2019/20

Algorithmische Bioinformatik
Übungsblatt 1

Ausgabe: 08. Oktober 2019 · **Besprechung:** 15. Oktober

Aufgabe 1.1 Gegeben sei ein 62-elementiges Alphabet Σ, repräsentiert durch die Zeichen `0`–`9`, `A`–`Z`, `a`–`z` in dieser Reihenfolge, sowie

- die Gleichverteilung u auf Σ,
- die Verteilung p, die dem i-ten Zeichen eine Wahrscheinlichkeit proportional zu $1/i^2$ zuordnet. (Wir fangen bei 1 an zu zählen, das erste Zeichen ist also '0', das 62. Zeichen 'z'.)

Bearbeite die folgenden Aufgaben, und miss bei den Simulationsaufgaben die Zeit (z.B. mit dem Linux-Befehl `time`). Es ist nicht notwendig, simulierte Sequenzen auszugeben. Zum Testen können natürlich kürzere Sequenzen simuliert und ausgegeben werden.

1. Simuliere einen Text der Länge 100 Millionen Zeichen gemäß u.
2. Bestimme die Verteilung p explizit. Wie groß ist jeweils p_0, p_A, p_a, p_z ?
3. Simuliere einen Text der Länge 100 Millionen Zeichen gemäß p mit linearer Intervallsuche.
4. Bestimme die kumulative Verteilungsfunktion P zu p.
5. Simuliere einen Text der Länge 100 Millionen Zeichen gemäß p mit binärer Intervallsuche.
6. Bestimme die Alias-Tabelle zu p.
7. Simuliere einen Text der Länge 100 Millionen Zeichen gemäß p mit der Alias-Methode.

Aufgabe 1.2 Auf der Webseite zur Vorlesung lässt sich eine Datei mit allen codierenden Sequenzen (Genen) des *Escherichia coli*-Bakteriums herunterladen. Da immer drei DNA-Nukleotide eine Aminosäure codieren, sollte die Länge jeder Sequenz ohne Rest durch 3 teilbar sein.

1. Betrachte das Dateiformat ("FASTA"-Format). Jede Sequenz beginnt mit einer Header-Zeile, die mit `>` beginnt. In den Zeilen darunter ist die DNA-Sequenz bis zur nächsten Header-Zeile (oder bis zum Dateiende) gespeichert. Die Zeilenumbrüche innerhalb der Sequenz haben keine Bedeutung. Schreibe ein Programm, dass die Anzahl der Sequenzen zählt und die Länge jeder Sequenz ermittelt. Ferner soll das Programm verifizieren, dass die Länge jeder Sequenz durch 3 teilbar ist und dass jede Sequenz mit dem Codon (Tripel) ATG oder GTG beginnt (typische Startcodons).
2. Erweitere das Programm so, dass für jedes Codon (nichtüberlappendes Tripel) gezählt wird, wie häufig es insgesamt in der Datei auftritt. Igoriere dabei das erste Auftreten von ATG in jeder Sequenz. Normalisiere die Häufigkeiten so, dass eine Verteilung entsteht. Welches ist das häufigste Codon? Welche der 64 möglichen Codons haben Wahrscheinlichkeit exakt Null bzw. fast Null und warum?
3. Schreibe nun ein Programm, das zufällige Gene an Hand der ermittelten Verteilung für eine vorgegebene Länge simuliert. Beginne dazu mit ATG und ziehe dann zufällige Codons aus der Verteilung (mit einer beliebigen Methode).