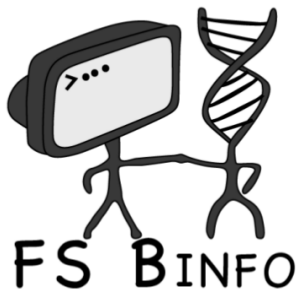


Weihnachtsübung



Hans Hawe
hans.hawe@campus.lmu.de

Fachschaft Bioinformatik

20. Dezember 2011

Zeitplan

- 25.10 Einführung in Linux und die Shell
- 08.11 Arbeiten mit Dateien auf der Shell
- 15.11 Shell 3 und Editor: vim
- 22.11 Einführung Java und Eclipse
- 29.11 Konditionen, Arrays und Schleifen
- 06.12 Rekursion und Iteration
- 13.12 File-I/O
- 20.12 **Weihnachtsübung**
- 10.01 Debuggen mit Eclipse
- 17.01 Objektorientierte Programmierung
- 24.01 Vererbung und Tipps
- 31.01 Quiz, Fragen und Evaluation

Webseite: <http://www.bioinformatik-muenchen.com/bioinfocom/informatik-tutorium>



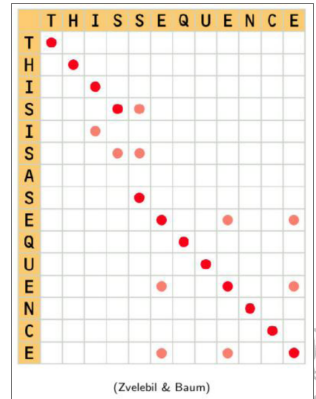
Bisher gelerntes

- Variablen und Arrays
- Kontrollstrukturen (if/else, for/while)
- Methoden definieren
- File I/O in Java
- und vielleicht noch ein bisschen mehr... :)



Thema heute: Dotplots

- Einfacher Sequenzvergleich
- Sequenzen an X- und Y-Achse
- Punkte indizieren Übereinstimmungen
- Ermöglicht identifizierung bestimmter Sequenzmerkmale



Programmierung: Anforderungen

- 2 Sequenzen als Input
- Berechnen der Hits
(speichern der Berechnungen in Matrix,...)
- Ausgabe des Dotplots auf STDOUT oder in Datei
- (...das ganze für mehr als nur 2 Sequenzpaare?..)



Programmierung: Konzept

- Benötigte Methoden und Variablen?
 - Speichern / Beschaffen der Sequenzen
 - Berechnung des Dotplos
 - Ausgabe des Dotplots
- Woher kommen die Sequenzen?
 - "hard-coded" im Programm
 - als Parameter
 - Auslesen aus Datei (z.B. FASTA-Datei)



Programmierung: Was wird benötigt?

- Variablen:
 - `static char hitSymbol;`
 - `static String seq1;`
 - `static String seq2;`
 - `static boolean[][] hitMatrix;`
- Methoden:
 - `calcDotplot();`
 - `printDotplot();` (evtl. überladen, "String fileName")



Programmierung:Umsetzung

- Implementieren benötigter Methoden
- Erstellen einer main-Methode
- Testen der Implementierung
- Erweiterung: readFastaSequences(String fileName)
- Wer kann/will: Objektorientierte Implementierung (z.B. Klasse Dotplot)
- ...
- .. z.B. mit NetBeans/Eclipse oder einfach mit nano/vi



Methode calcDotplot

```
GNU nano 2.2.6      File: main.simple.java      Modified

/**
 * Actually calculates the hitmatrix, i.e. makes the
 * needed sequence comparisons
 */
private static void calcDotplot() {
    hitMatrix = new boolean[seq1.length()][seq2.length()];
    for (int i = 0; i < seq1.length(); i++) {
        for (int j = 0; j < seq2.length(); j++) {
            // charAt gets the char of a String at position i/j
            if (seq1.charAt(i) == seq2.charAt(j)) {
                hitMatrix[i][j] = true;
            } else {
                hitMatrix[i][j] = false;
            }
        }
    }
}

^G Get Help  ^O WriteOut ^R Read Fil ^Y Prev Pag ^K Cut Text ^C Cur Pos
^X Exit
```

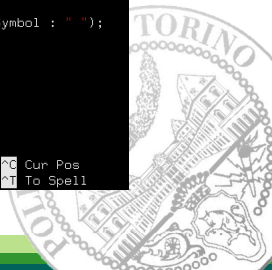


Methode printDotplot

```
GNU nano 2.2.6      File: main.simple.java

/**
 * Prints the calculated plot to STDOUT
 */
private static void printDotplot() {
    // check whether data has been calculated
    for (int i = -1; i < seq1.length(); i++) {
        StringBuilder sb = new StringBuilder();
        for (int j = -1; j < seq2.length(); j++) {
            if (i == -1) {
                if (j == -1) {
                    sb.append(" ");
                } else {
                    sb.append(" ").append(seq2.charAt(j));
                }
            } else {
                if (j == -1) {
                    sb.append(seq1.charAt(i));
                } else {
                    sb.append(" ").append(hitMatrix[i][j] == true ? hitSymbol : " ");
                }
            }
        }
        System.out.println(sb.toString());
    }
}

^G Get Help      ^O WriteOut     ^R Read File    ^Y Prev Page    ^K Cut Text      ^C Cur Pos
^X Exit          ^J Justify      ^W Where Is    ^V Next Page    ^U UnCut Text   ^T To Spell
```



Klasse main

```
GNU nano 2.2.6      File: main.simple.java      Modified
public class main {
    // some needed vars
    private static String seq1 = null;
    private static String seq2 = null;
    private static boolean[][] hitMatrix;
    private static char hitSymbol = 'x';

    public static void main(String[] args) {
        // check for arguments argument and assume sequences
        if (args.length == 2) {
            seq1 = args[0];
            seq2 = args[1];
        } else {
            // just create dotplot for two random strings
            seq1 = "SOMERANDOMSEQUENCE";
            seq2 = "SOMERANDSEQUENCE";
        }
        // create and print plot
        calcDotplot();
        printDotplot();
    }
}
```



Programmierung: Erweiterung

- Sequenzen "hard-gecoded" in main
- Erweiterung:
 - Lesen von Sequenzen aus FASTA-Datei
 - Paarweiser Dotplot aller Sequenzen

```
1 >sp|P00750|TPA_HUMAN Tissue-type plasminogen activator O5=Homo sapiens GN=PLAT PE=1 SV=1
2 MDAMKRGKCCVLLLCGAVFVSPSQEIHARFRRGARSYQVICRDEKTMIIYQQHQSWLRPV
3 LRSNRVEYCWNSGRAQCHSVPVKSCEPRCFNGGTCQQALYFSDFCVQCPEGFAGKCCCE
4 IDTRATCYEDQGISYRGTWSTAESGAECTNWNSSALAQKPYSGRRPDAIRLGLGNHNYCR
5 NPDRDCKPWCYVFKAGKYSSEFCSTPACSEGNDCYFNGNSAYRGTHSLTESGAFCLPWN
6 SMILIGKVYTAQNPSAQLGLGKHNYCRNPDGDAKPWCHVLKNRRLTWEYCDVPSCSTCG
7 LRQYSQPQFRIKGGLFADIASHPWQAAIFAKHRRSPGERFLCGGILISSCWLSAAHCFQ
8 ERFPFHLLTVILGRTYRVVPGEEEQKFEVEKYIVHKEFDDDTYDNDIALLLQKSDSSRCA
9 QESSVVRTVCLPPADLQLPDWTECELSGYGKHEALSPFYSERLKEAHVRLYPSSRCTSQH
10 LLNRTVTDNMLCAGDTRSGGPQANLHDACQGD5GGPLVCLNDGRMTLVGIIISWGLCGCQK
11 DVPGVYTKVTNYLDWIRDNMRP
12 >sp|Q85Q23|TPA_PIG Tissue-type plasminogen activator O5=Sus scrofa GN=PLAT PE=2 SV=1
13 MYALKRELWCVLLLCGAICTSPSQETHRRLRRGVRSYRVTCRDEKTMIIYQQHQSWLRPL
14 LRGNRVEHCWCNDGQTQCHSVPVKSCEPRCFNGGTCQAIYFSDFCVQCVPVGFIGRQCE
15 IDARATCYEDQGITRYGTWSTTESGAECVNWNTSGLASMPYNGRRPDAVKLGLGNHNYCR
16 NPDKDCKPWCYIFKAEKYSPDFCSTPACTKEKEECYTGKGLDYGRTSLTMSGAFCLPWN
17 SLVLMGKIYTAWNSNAQTLGLGKHNYCRNPDGDTQPWCHVLKDHKL TWEYCDLPQCVTGCG
18 LRQYKEPQFRIKGGLYADITSHPWQAAIFVKHRRSPGERFLCGGILISSCWLSAAHCFQ
19 ERFPFHVHVVLGRTYRLVPGEEEQAFVEVEKYIVHKEFDDDTYDNDIALLLQKSDSLTCA
20 QESDAVRTVCLPEANLQLPDWTECELSGYGKHEASSPFYSERLKEAHVRLYPSSRCTSKH
21 LFNKTIINMLCAGDTRSGGDNANLHDACQGD5GGPLVCMKGNHMTLVGVISWGLCGCQK
22 DVPGVYTKVTNYLNWIRDNTRP
```



Programmierung:Erweiterng

- FASTA-Datei als Parameter des Programms (in String[] args...)
- Neue Methode "readFastaSequences" in main
- Liefert alle Sequenzen der FASTA-Datei (als String[], ArrayList<String>etc...)
- Erstellen eines Dotplots aller Sequenzen untereinander
 - z.B. mit Hilfe zweier for-Schleifen...



Klasse main (erweitert)

```
GNU nano 2.2.6      File: main.java
import java.io.BufferedReader;
import java.io.FileReader;
import java.io.IOException;
import java.util.ArrayList;

public class main
{
    // some needed vars
    private static String seq1 = null;
    private static String seq2 = null;
    private static boolean[][] hitMatrix;
    private static char hitSymbol = 'x';

    public static void main(String[] args) {
        // check for first argument
        if (args.length == 1) {
            try {
                // get list of sequences and create all pairwise dotplots
                ArrayList<String> sequences = readFastaSequences(args[0]);
                for (int i = 0; i < sequences.size(); i++) {
                    for (int j = i + 1; j < sequences.size(); j++) {
                        // create and print dotplots
                        seq1 = sequences.get(i);
                        seq2 = sequences.get(j);
                        calcDotplot();
                        printDotplot();
                        // add a delimiter line to distinguish between plots
                        System.out.println("-----\n");
                    }
                }
            } catch (IOException e) {
                e.printStackTrace();
            }
        } else if (args.length == 2) {
            // check for arguments argument and assume sequences
            seq1 = args[0];
            seq2 = args[1];
            calcDotplot();
            printDotplot();
        } else {
            // just create dotplot for two random strings
            seq1 = "S08ERAND08SEQUENCE";
            seq2 = "S08ERAND08SEQUENCE";
            calcDotplot();
            printDotplot();
        }
    }
}

Wrote 128 lines
```



Klasse main: readFastaSequences

```
GNU nano 2.2.6          File: main.java          Modified

/**
 * Reads in a single list of strings from a fasta file
 * (does not map sequences to their identifier)
 * @param fileName The file from which to read the sequences
 * @return A list of read strings or an empty list if no strings could be read
 * @throws IOException
 */
public static ArrayList<String> readFastaSequences(String fileName) throws IOException {
    // create needed list for read strings
    ArrayList<String> out = new ArrayList<String>();
    // open the file for reading
    BufferedReader reader = new BufferedReader(new FileReader(fileName));
    // prepare vars
    String line = null;
    StringBuilder sb = new StringBuilder();
    // iterate through all lines of the file
    while ((line = reader.readLine()) != null) {
        // check whether we got a sequence header
        if (line.startsWith(">")) {
            if (sb.length() > 0) {
                // save the current line if any
                out.add(sb.toString());
                sb = new StringBuilder();
            }
        } else {
            // just append line
            sb.append(line);
        }
    }
    // add the last string
    out.add(sb.toString());
    // return list of strings
    return out;
}

␣␣
⌘ Get Help      ⌘ WriteOut     ⌘ Read File    ⌘ Prev Page    ⌘ Cut Text     ⌘ Cur Pos
⌘ Exit          ⌘ Justify      ⌘ Where Is    ⌘ Next Page    ⌘ UnCut Text   ⌘ To Spell
```



Für zu Hause

- Überladen der "printDotplot"-Methode
 - Parameter "String fileName", "boolean append"
 - Schreiben des Dotplots in die angegebene Datei
 - Inhalt der Datei bei Bedarf ("append==true") nicht überschreiben sondern neuen Inhalt hinzufügen
- Ausgabe, ob beide Sequenzen vollkommen identisch
- Fortgeschritten: Erzeugen eines graphischen Outputs/Bildes für den Dotplot



Ende

Frohe Weihnachten!

