Einführung in die Einführung in die Angewandte Bioinformatik: Organisation, Allgemeines, Unix 16.04.2009

Prof. Dr. Sven Rahmann

Team

Prof. Dr. Sven Rahmann (Vorlesung)
Dipl.-Inform. Marcel Martin (Übungen)

Zeit Do 12-14; Übungen um 14, 15, 16, 17 Uhr
Ort Vorlesung in der Chemie HS 3; Übungen in OH14, U04 (Keller)

Alle Informationen

Webseite zur Vorlesung: http://bioinfo.wikidot.com/

Sprechstunde von Prof. Rahmann

Mo 16-17 in OH14, R214
Bitte möglichst per e-mail anmelden, sonst evtl. sehr lange Wartezeiten!
Sven.Rahmann /at/ tu-dortmund.de

Wichtiger Hinweis!

Die Vorlesung nächste Woche (23.04.)
muss aus organisatorischen Gründen leider ausfallen.

Aber:
Die Übung am 23.04. findet statt!
Nutzen Sie die Zeit, um die Grundlagen von heute zu üben!

Danach (ab 30.04.) ist für solche Fragen keine Zeit mehr:
- Wie starte ich meinen Browser?
- Was ist ein Shell-Fenster?
- Wie starte ich R?

Ziele der Vorlesung und Übungen

Wissen über

- Aufgaben,
- Methoden,
- Beschränkungen der Bioinformatik

Anwendungskenntnisse zu

- bioinformatischen Datenbanken
- im WWW verfügbarer Software

Einführung in Methoden der Informatik (sehr knapp)
Ziel dabei: Wie kann ich mit Informatikern reden,
biologische und chemische Fragestellungen/Probleme formulieren?

Prüfungsleistung

- Klausur und praktische Prüfung (am Übungs-PC)
  am 23.07.2009 in der jeweiligen Übungsgruppe

- Nehmen Sie unbedingt regelmäßig an den Übungen teil !

- Normale Übungsaufgaben enthalten zahlreiche Hinweise.
- Es genügt nicht, diese „abzuarbeiten“ !
- Sie sollen verstehen, was Sie tun, wenn Sie eine Aufgabe lösen !!
- Manche Aufgaben enthalten keine Hinweise, simulieren Klausurbedingungen.
- Diese geben Ihnen eine Rückmeldung zu Ihrem Kenntnisstand.

Literatur (Empfehlungen)

P.M. Selzer, R. J. Marhöfer, A. Rohwer (2004)
Angewandte Bioinformatik – Eine Einführung
Springer-Verlag

Jean-Michel Claverie, Cedric Notredame (2006)
Bioinformatics for Dummies, 2. Auflage
Wiley & Sons

Nello Christianini and Matthew W. Hahn (2007)
Introduction to Computational Genomics – a Case Studies Approach
Cambridge University Press

D.W. Mount (2004)
Bioinformatics: Sequence and Genome Analysis, 2. Auflage
Cold Spring Harbor Laboratory Press

Biologie: bio = Leben, logos = Wissenschaft
Biologie: Wissenschaft des Lebens
Biologie früher: Katalogisieren von Lebensformen
Biologie heute: molekular geprägt (seit der Entdeckung der DNA)
Basis der modernen Biologie: Chemie

Informatik: Wissenschaft der systematischen Verarbeitung von Information
Information: Ordnung, Struktur, Abweichung vom Zufall

Wie passt das zusammen?
Entstehung von Leben = Bildung von Ordnung / Strukturen [?]

Z.B. sind Zellen vor allem damit beschäftigt, die innere Ordnung zu erhalten.
Lebewesen bleiben am Leben, weil sie sich von ihrer Umwelt abgrenzen.

Andere Bio-x - Wissenschaften

Bio-x bedeutet eins von zwei Dingen:
[Hier ist *x* ein Platzhalter (Variable)]

1. Wissenschaft *x* leistet Beitrag zum besseren Verständnis der Biologie
2. Biologie inspiriert neue Forschungsrichtungen in Wissenschaft *x*

Beispiele:

- Biochemie
- Biophysik
- Biotechnologie
- Biomathematik
- Bioinformatik

Bioinformatik ist ein weites Feld...

Bioinformatik, Systembiologie: Modewörter der letzten 20 / 5 Jahre

Bioinformatik = mehrere Disziplinen (von theoretisch bis angewandt):
Biomathematik

- Theoretische Biologie
- (Theoretische) Ökologie
- Biostatistik
- Sequenzanalyse
- „Computational biology“
- Bioinformatik (im engeren Sinn)
- Systembiologie
- Computational *omics: genomics, transcriptomics, proteomics, ...
  [*omics: Untersuchung der Gesamtheit von *; der Stern * ist ein Platzhalter]
- Angewandte = praktische Bioinformatik

Bioinformatiker und Anwender

Bioinformatiker

Person, die Modelle, Methoden und Programme
aus der Informatik und Mathematik
entwickelt und anwendet,
um Fragestellungen aus den molekularen Lebenswissenschaften zu lösen.

Bioinformatik-Anwender

wie oben, ohne „Modelle“ und „entwickelt“.

Wichtig

In dieser Vorlesung werden Sie zu einem (gut informierten) Anwender.
Sie lernen **nicht**, formale Modelle zu entwickeln.
Sie lernen auch nicht programmieren (außer ein wenig R).

Tipp: Besuchen Sie später evtl. einen Programmierkurs.

Informatik in der Biologie

Hauptgrund: Hochdurchsatztechnologien, große Datenmengen
(z.B. DNA-Sequenzierung, Massenspektrometrie, Mikroskopie-Aufnahmen)

Verwaltung von großen Datenmengen

Datenbanken (schneller Zugriff, Auffindbarkeit von Informationen)
Ausfall-tolerante Systeme (z.B. auch bei Festplattencrash)

Analyse von großen Datenmengen

z.B. Genomassemblierung, Identifikation von Metaboliten,
hierzu braucht man effiziente Algorithmen und gute Hardware!

Design von Experimenten

Maximum an neuen Informationen mit möglichst wenig Aufwand?
Welche Untersuchung ist am Erfolg versprechendsten?

Simulation

Vermeidung von (teuren) Experimenten – Vorhersage am Computer
(z.B. Wirkungsweise eines neuen Medikaments anhand molekularer Dynamik)

Begriffserklärungen

Hardware

Oberbegriff für die maschinentechnische Ausrüstung eines Systems - („kann man anfassen“).

Software

Zusammenfassender Begriff für Programme und Daten in Computern - Dazu gehören Betriebssysteme und Anwendungsprogramme.

Betriebssystem (operating system, OS)

Software, die die Verwendung (den Betrieb) eines Computers ermöglicht.
Verwaltet Betriebsmittel wie Speicher, Ein- und Ausgabegeräte.
Steuert die Ausführung von Programmen.
Beispiele: Linux, Unix, Solaris, Windows, Mac OS, ...

Begriffserklärungen

Browser (to browse: durchstöbern)
Programm, um Informationen übersichtlich zu betrachten, z.B.
- Dateibrowser zeigt Dateien in einem Verzeichnis an.
- Webbrowser zeigt Seiten aus dem Internet an.

Account, home, login, password
Solaris (das Betriebssystem der Firma Sun) und Linux: Mehrbenutzersysteme.
Jeder hat seinen eigenen Arbeitsbereich (home-Verzeichnis).
Zugriffsbeschränkung durch eigene Benutzerkennung (account),
dazu geheimes Passwort.

Verzeichnis (directory, Ordner)
„Behälter“ für verschiedene Dateien.
Man benutzt eine Hierarchie von Verzeichnissen, um Ordnung zu halten.
So entsteht ein Verzeichnisbaum.

Begriffserklärungen

Arbeitsumgebung und Fenster-Manager (z.B. KDE mit KWin)

- Grundlegende Funktionen für die Arbeit mit dem Computer
- Grundlegende Fensterfunktionen (Minimieren, Vergrößern, Schließen)

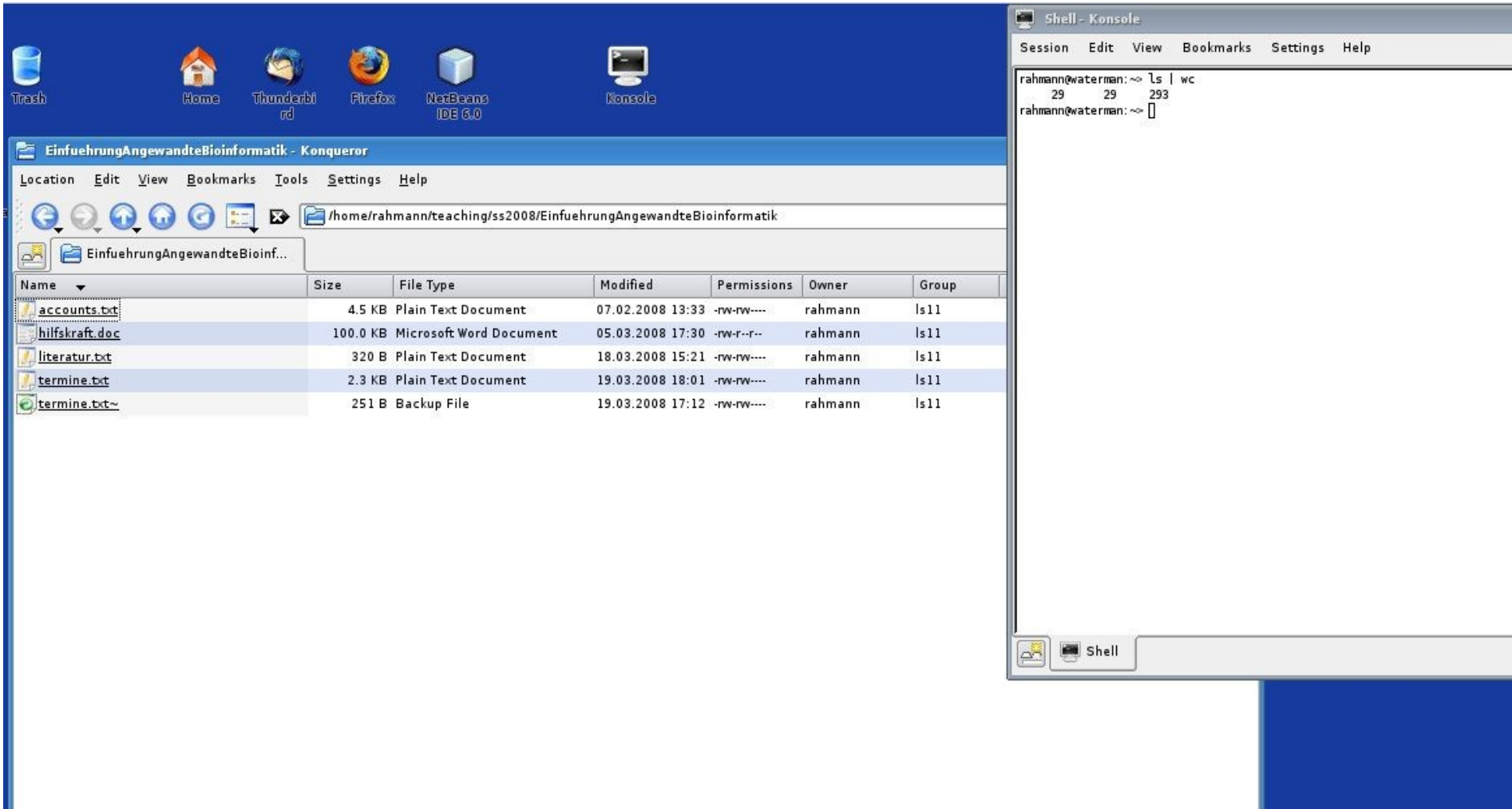
GUI (graphical user interface, graphische Benutzeroberfläche)

erlaubt das Ausführen von Aktionen (z.B. Verzeichnis anlegen)
mit Hilfe von Menüs und Dialog-Fenstern

Shell

Textorientierte Eingabe-Schnittstelle zwischen Computer und Benutzer,
erlaubt die Eingabe von Befehlen (z.B. mkdir zum Anlegen eines Verzeichnisses);
zunächst weniger komfortabel, aber mächtiger als ein GUI.

Beispiel:
KDE-Desktop mit Dateibrowser (Konqueror) und Shell-Fenster



Arbeiten mit der Shell

In einem Shell-Fenster („Konsole“) können Befehle eingegeben werden.

Generelle Befehlssyntax

Befehl Optionen Argumente

- „Befehl“ ist der Name des Befehls
  Beispiel: `ls` listet alle Dateien in einem Verzeichnis auf
- „Optionen“ modifizieren das Verhalten eines Befehls;
  beginnen mit Minus.
  Beispiel zu `ls`: `-a` zeigt alle Dateien (auch sonst versteckte, „all“),
  `-l` zeigt ausführliche Informationen („long“)
- „Argumente“ geben an, worauf der Befehl angewendet werden soll;
  häufig der Name eines Verzeichnisses oder einer Datei (oder mehrere)
  Beispiel: Ein Punkt (.) steht für das aktuelle Verzeichnis.

Gesamtbeispiel: `ls -l -a .`
Kürzer: `ls -la` (Optionen kombiniert; Punkt ist hier Standard, weggelassen)

Verzeichnis-Befehle

- `pwd`: aktuelles Verzeichnis anzeigen
- `mkdir`: neues Verzeichnis anlegen (`mkdir testvz`)
- `cd`: Verzeichnis wechseln (`cd testvz`)
  `cd` ohne Verzeichnisnamen wechselt in Ihr home-Verzeichnis
  Ein Punkt (.) steht für das aktuelle Verzeichnis;
  zwei Punkte (..) für das Verzeichnis darüber.
- `rmdir`: Verzeichnis entfernen, wenn leer (`rmdir testvz`)
  Warnung: Löschen kann nicht rückgängig gemacht werden
- `ls`: Verzeichnisinhalt anzeigen
- `ls -la`: Verzeichnisinhalt detailliert anzeigen

Beispiel: Was passiert jeweils, wenn Sie nacheinander

```
cd; pwd; mkdir uebung; cd uebung; pwd;
cd ..; rmdir uebung; ls; pwd
```

eingeben?

Hinweis zu Übungsaufgaben

Es ist sinnvoll, jeden Übungszettel in einem eigenen Verzeichnis zu bearbeiten, z.B. `uebung1`, `uebung2`, ...
Für die anfallenden Verwaltungsaufgaben
(Anlegen, Umbenennen, Löschen von Verzeichnissen) können Sie:

- einen Dateibrowser (Konqueror) verwenden,
- die Shell (Konsole) verwenden und Befehle eingeben.

Die Wahl steht Ihnen frei.
Versuchen Sie aber, die Shell-Befehle zu üben!

Philosophie der Unix-Befehle

Jeder Befehl führt eine einfache, klar umrissene Aufgabe aus.
Reichhaltige Optionen ermöglichen viele Variationen.
Komplexe Aufgaben werden durch Aneinanderreihung von Befehlen möglich.

Hilfe zu Befehlen und Optionen anzeigen lassen

`man` befehlsname („manual pages“)
z.B. `man ls` zeigt die Wirkung und alle möglichen Optionen von ls

Datei- und Verzeichnisbefehle

`cp Datei1 Datei2` – kopiert Datei1 nach Datei2
`cp Dateiliste Verzeichnis` – kopiert alle Dateien der Liste ins Verzeichnis
`mv Datei1 Datei2` – wie `cp`, aber löscht Original (move, Umbenennung)
`rm Dateiliste` – löscht alle in der Liste angegebenen Dateien
Warnung! Löschen mit `rm` oder rmdir kann nicht rückgängig gemacht werden!

Verwendung von wildcards * und ?

Häufig darf man statt einer einzelnen Datei eine Liste von Dateien angeben.
Statt diese explizit aufzulisten, kann man * und ? verwenden:

- * steht für eine beliebige Folge von Zeichen
- ? steht für ein beliebiges Zeichen

z.B. `rm test*.fasta`
löscht alle Dateien, deren Name mit test anfängt und deren Endung .fasta ist.

Vorsicht: Niemals `rm *` ausprobieren!
Löscht alles im aktuellen Verzeichnis!

Untersuchung von Dateien

- `cat Dateiliste`
  zeigt nacheinander die Inhalte aller Dateien an

- `head -N 17 Dateiliste`
  zeigt jeweils die ersten 17 Zeilen an

- `tail -N 17 Dateiliste`
  zeigt jeweils die letzten 17 Zeilen an

- `more Dateiliste`:
  zeigt nacheinander die Inhalte aller Dateien an,
  wartet auf Tastendruck wenn der Bildschirm voll ist

Untersuchung von Dateien – grep und wc

grep (general regular expression matcher)

`grep Muster Dateiliste`
sucht nach Muster in allen Dateien,
gibt alle Zeilen aus, in denen das Muster auftritt.
Beispiel: `grep Meier telefonbuch.txt`
(Datei ist Telefonbuch, ein Eintrag mit Name + Telefonnummer pro Zeile).
Sucht alle Einträge mit Namen „Meier“.
Das Muster kann viel komplizierter sein (reguläre Ausdrücke), z.B.
`grep M..er telefonbuch.txt`
Hier steht der Punkt für ein beliebiges Zeichen, man findet Meier, Mayer, Maler, ...

wc (word counter)

`wc Dateiliste`
gibt für jede Datei 3 Zahlen aus: Anzahl Zeilen, Wörter, Zeichen

Ein- und Ausgabeumleitung, Pipes (<, >, |)

Ausgabeumleitung >

Das Zeichen > leitet die Ausgabe eines Befehls in eine Datei um.
Achtung: Wenn die Datei schon existiert, wird sie überschrieben!

```
grep Meier telefonbuch.txt > meiers.txt
```

Eingabeumleitung <

Analog kann man die Eingabe zu einem Programm
aus einer Datei (statt z.B. von der Tastatur) beziehen.

Pipe |

Will man die Ausgabe eines Programms als Eingabe eines anderen verwenden,
kann man die Pipe (Rohr) | benutzen:

```
ls | wc
```

Zählt, wie viele Dateien im aktuellen Verzeichnis sind.
Die Ausgabe von `ls` (Verzeichnisinhalt) wird als Eingabe für `wc` verwendet.
Die Ausgabe von `wc` erscheint im Shell-Fenster.

Das FASTA-Sequenz-Dateiformat

Biologische Sequenzen (DNA, Proteine)
weden gerne im sog. FASTA-Format gespeichert.
Dabei können sie in einer „Titelzeile“
mit zusätzlichen Informationen annotiert werden.

Eine Datei kann aus mehreren Sequenzen bestehen.
Jede Titelzeile (header) beginnt dabei immer mit `>`.
Esfolgen Sequenzdaten bis zum nächsten header oder bis zum Dateiende.

Beispiel (2 Sequenzen)

```
>Lieblingssequenz
ACGTTGCA
>andere Sequenz aus dem Internet
AAAAAAAAAA
AAAAAAAAAA
AAAAAAAAAT
```