Bioinformatik

Für Biophysiker

Sommersemester 2009

Silke Trißl / Ulf Leser

Wissensmanagement in der Bioinformatik

Wissensmanagement in der Bioinformatik

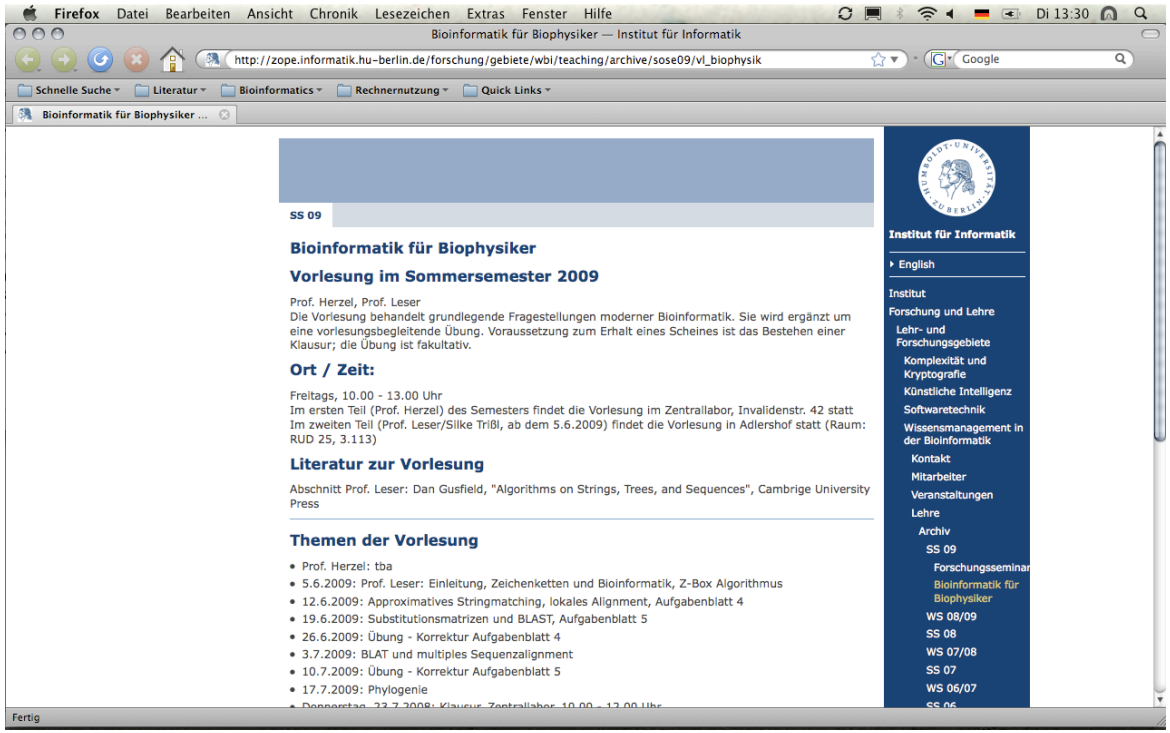
- Schwerpunkte
  - Algorithmen der Bioinformatik
  - Management molekularbiologischer Daten
  - Datenintegration
  - Text Mining

Mein Teil

- Algorithmen der Bioinformatik
- Vorlesung
  - 5 Termine a 3 Stunden
  - 5.6., 12.6., 19.6., 3.7., 17.7.
- Übung
  - 2 Termine a 3 Stunden
  - 26.6.
  - 10.7.
- Klausur
  - Am Donnerstag, 30.7.2008, 10.00 Uhr, Zentrallabor

Informationen

- Zu Vorlesung und Praktikum
  - http://www.informatik.hu-berlin.de/wbi



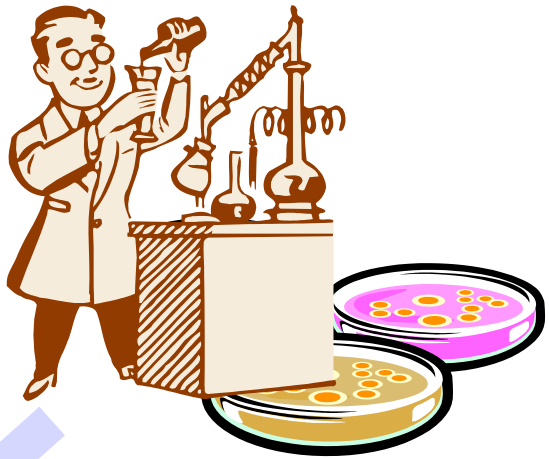
- Bei Fragen
  - trissl(at)informatik.hu-berlin.de

Literatur

- Primär
  - Dan Gusfield: „Algorithms on Strings, Trees, and Sequences“, Cambridge University Press, 1997 (ca. 60 Euro)
- Weitere
  - Joachim Böckenhauer, Dirk Bongartz: „Algorithmische Grundlagen der Bioinformatik“, Teubner, 2003 (ca. 30 Euro)

Bioinformatik – was ist das?

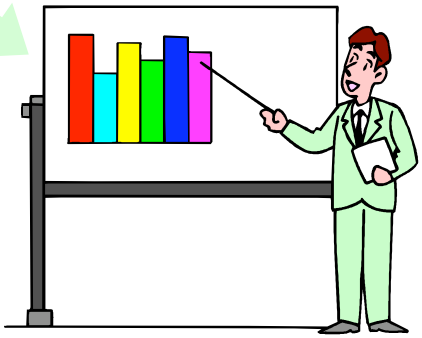
- ein relativ junges Fachgebiet
- viele biologische und medizinische Daten



wiederfinden

speichern

analysieren



präsentieren

Algorithmen

- Algorithmus = Folge von Anweisungen zur Lösung eines Problems

- Beschreibung eines Algorithmus
  - unabhängig von der Programmiersprache oder Rechnerarchitektur
  - Komplexität der Laufzeit
    - wie oft muss ich die Anweisungen für eine Eingabe ausführen
  - Komplexität des Speicherplatzes
    - wie groß können Zwischenergebnisse werden

Wichtiger Begriff: Komplexität

- Komplexität wird gemessen in der Länge der Eingabe n
  - Anzahl Zahlen, die es zu sortieren gilt
  - Anzahl Knoten eines Graphen, den es zu durchsuchen gilt
  - Länge der Sequenzen, die es zu vergleichen gilt
  - ...
  - [Nicht immer einfach zu bestimmen]

- Bei uns meistens
  - Anzahl und Länge von Zeichenketten (DNA/AA)

- Komplexität wird durch die O-Notation ausgedrückt

Beispiel: Komplexität

- Beispiel
  - In einer Liste von n Zahlen solle jede Zahl verdoppelt werden
- Algorithmus

```
für jede Zahl i
   berechne i*2
```

- Komplexität
  - Laufzeit: O(n)
  - Speicherplatz: O(n)

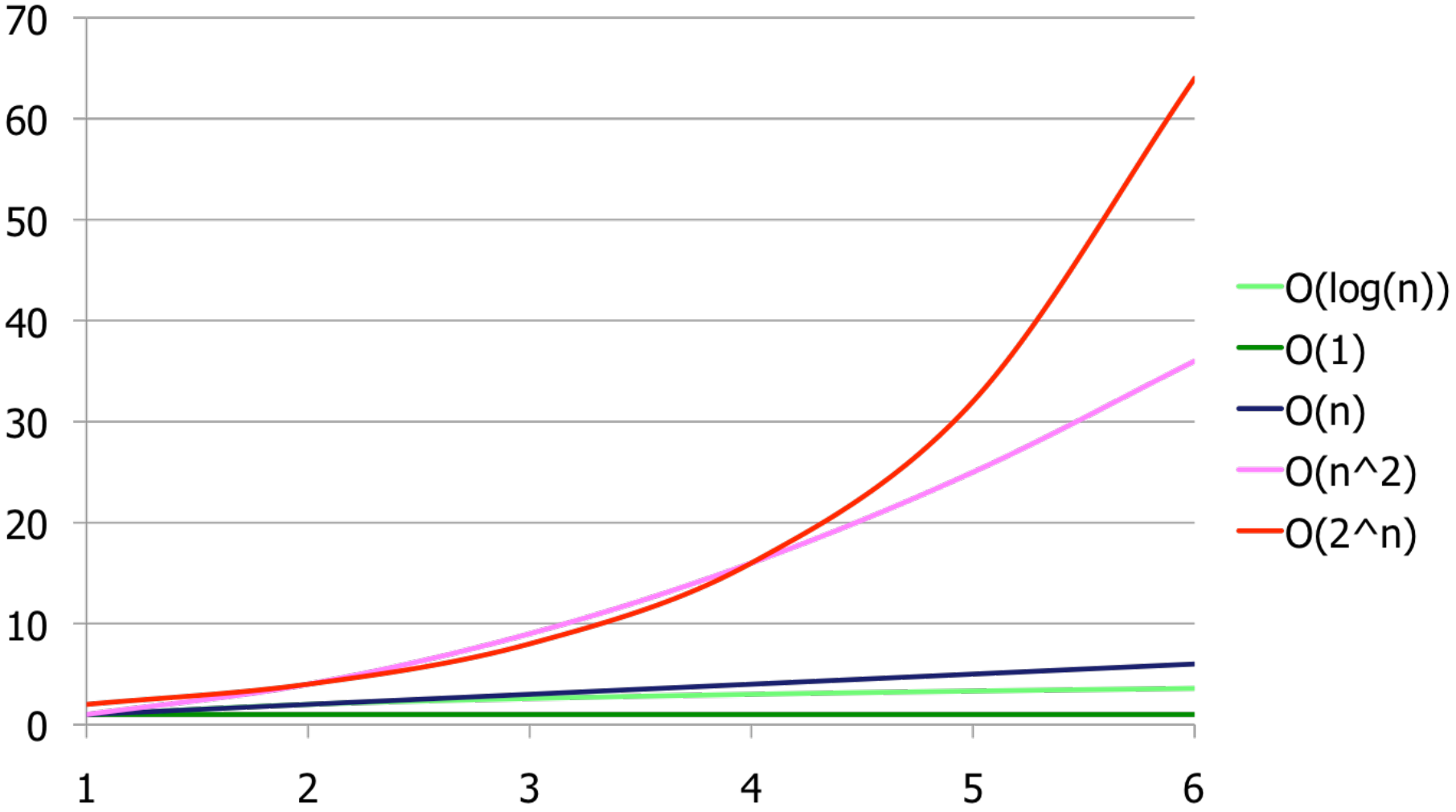
O-Notation

- Komplexität wird angegeben als O(g(n))
- Das hat folgende Bedeutung
  - Algorithmus A ist O(g(n)) gdw. es eine Funktion f(n) gibt, die die Laufzeit von A berechnet, und es gilt:
    - Es gibt positive Konstanten K und n0 mit:
    - $|f(n)| \leq K*|g(n)|$ für alle $n \geq n0$
- Beispiel:
  - $f(x) = 6x^4$ ➔ $O(x^4)$

O-Notation

- Beispiel
  - $8n^3 + n^2 + 76$ ist $O(n^3)$
  - Da $8n^3 + n^2 + 76 \leq 85n^3$ für alle $n \geq 1$
- Für g wählt man i.d.R. nur den Grad des Polynoms bzw. den höchsten Exponenten
  - $O(1)$, $O(n)$, $O(n^2)$, $O(n^3)$, $O(2^n)$, $O(\log(n))$, ...
- Ein Großteil der Informatik widmet sich
  - der Suche nach Algorithmen mit geringerer Komplexität
  - Der Suche nach besten Algorithmen für Klassen von Problemen

Was bedeutet welche Komplexität?



Take Home Message

- Informatiker lieben O-Notation
- Wichtig für alle Anwendungen am Computer, die
  - mit nicht-trivialen Problemen oder
  - mit nicht-trivialen Datenmengen zu tun haben
- Kleine O sind gut; n im Exponenten ist schlecht
- Schon $O(n^2)$ ist für große Datenmengen zu viel
  - Beispiel: Sequenzalignment ist O(n*m)
  - Längstes Proteine des Menschen in SwissProt
    - Mucin-16 (Ovarian carcinoma antigen CA125)
    - Länge: 22152 AA (repeats, duplication)
    - Entspricht 66456 DNA basen
  - Vergleich mit entsprechendem Gen in Maus
    - Erfordert 4.416.399.936 Basenvergleiche
  - Suche nach dem ähnlichsten Gen in Maus (ca. 22000 Gene, avg~1200 Basen)
    - Erfordert ~22.000*(1200*66456)= 1.754.438.400.000

Themen der Vorlesung

1. Exaktes Stringmatching
2. Approximatives Stringmatching
3. Heuristische Suche
4. Multiples Sequenzalignment
5. Phylogenie

1. Stringalgorithmen

- Gegeben ein Template T und ein Pattern P. Finde alle Vorkommen von P in T in möglichst kurzer Zeit
  - Exaktes Matching

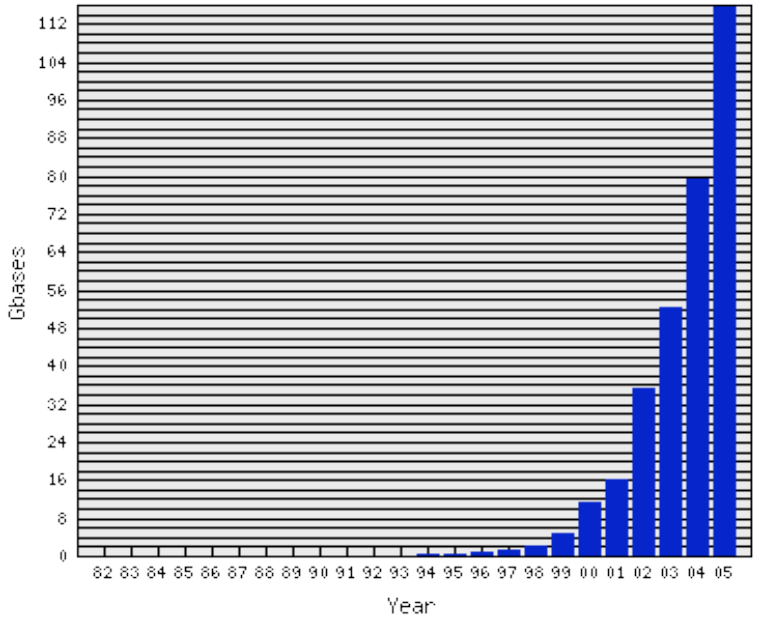
- Naive Suche
- Z-Box Fundamentaler linearer Algorithmus
- Boyer-Moore Schnellster Algorithmus in der Praxis
- Varianten
  - Suche nach mehreren P
  - Suche mit regulären Ausdrücken (= endlichen Automaten)

2. Approximatives Stringmatching

- Approximatives Matching
  - Gegeben zwei Sequenzen T und P. Wie ähnlich sind sich P und T?
- Variante
  - Gegeben zwei Sequenzen T und P. Welches sind die Substrings in T, die „ähnlich" zu einem Substring in P sind
- Beides sind fundamentale Fragestellung der Bioinformatik
  - Globales versus lokales Alignment
  - Ähnlich Sequenz – ähnliche Struktur – ähnliche Funktion
- Was heißt überhaupt ähnlich?
  - Edit-Abstand, Alignierung
- Naiver Algorithmus benötigt exponentielle Laufzeit
  - Verbesserung durch tabellarische Berechnung

3. Heuristiken zur approx. Suche

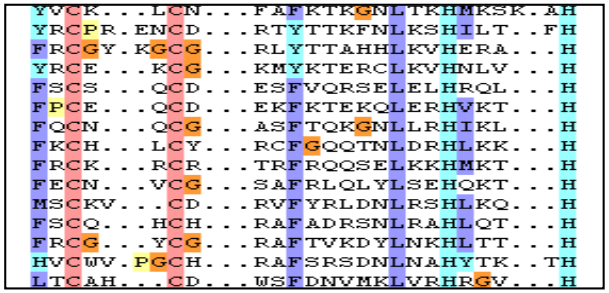
- Quadratische Laufzeit ist zu teuer
  - Genomanalyse benötigt Suche auf allen bekannten Sequenzen
  - Celera Sequenzierung: All-against-all Vergleich von 28.000.000 Teilsequenzen



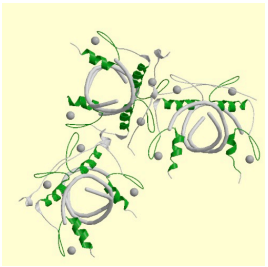
- Grundidee von BLAST
  - In jedem guten approximativen Match steckt ein exakter Match
  - Mischung von exaktem und approximativen Match
  - Findet nicht alle Hits, aber die „meisten"

4. Multiples Alignment

- Gegeben eine Menge von Strings. Ein Multiple Sequence Alignment (MSA) ist eine Anordnung der Strings mit Spaces untereinander



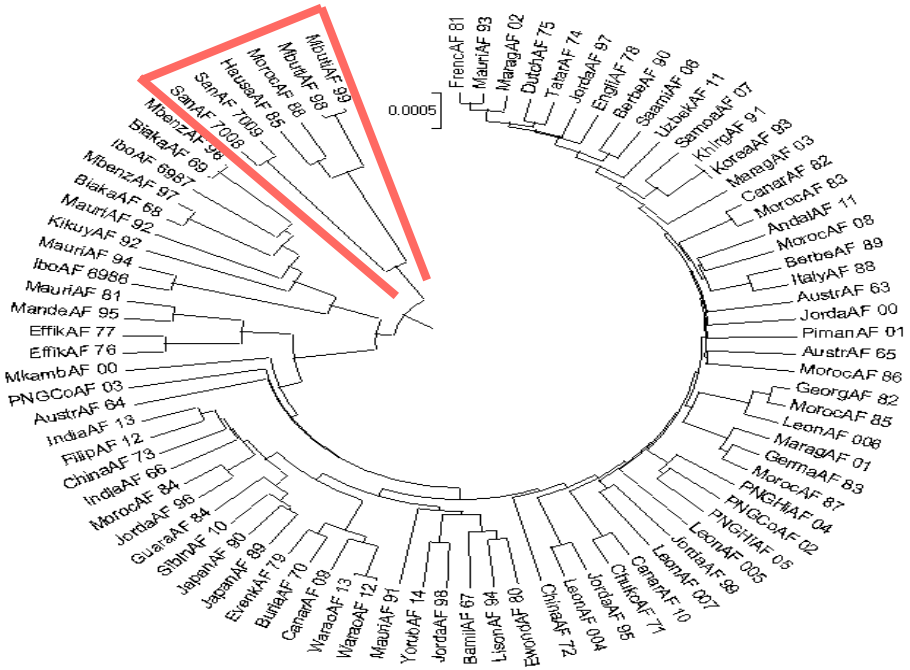
Quelle: Pfam, Zinc finger domain

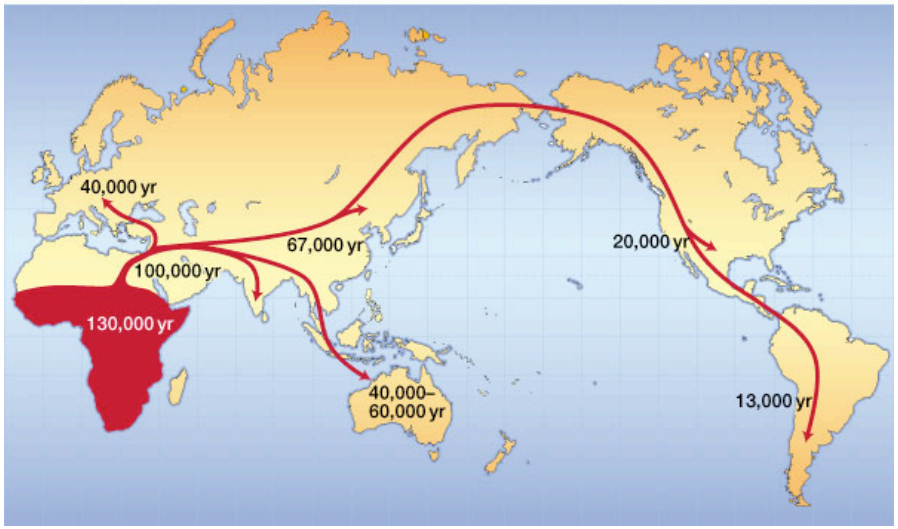


- Ziel: Finde das „Gemeinsame" der Sequenzen
  - Funktionen werden oft von sehr kurzen Sequenzstücken bestimmt
  - Welcher Teil eines Proteins bestimmt die Funktion?
  - Wie kann man Proteine in Familien anordnen?

5. Phylogenetische Algorithmen

- Sequenzierung der mitochondrialer DNA (16 KB) von 86 geographisch verteilt lebenden Personen
- Ergebnis: Mitochondriale DNA scheint nach einer molekularen Uhr abzulaufen; Divergenz ist ca. 1,7E-8 pro Base und Jahr





Quelle:
Ingman, M., Kaessmann, H., Pääbo, S. & Gyllensten, U. (2000)
Nature 408: 708-713

Quelle:
http://www.genpat.uu.se/mtDB/sequences.html
Methode: UPGMA