

Bioinformatik

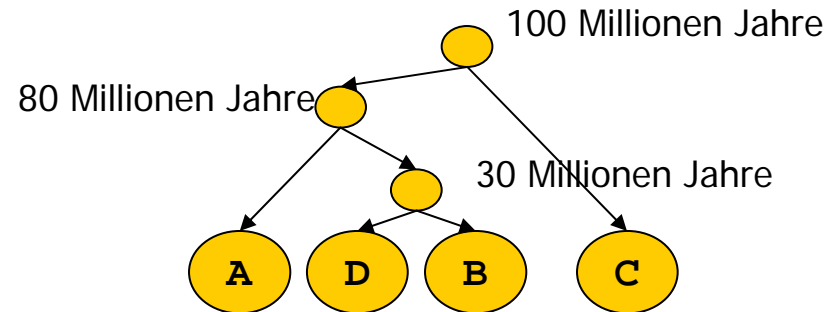
Character-basierte Verfahren
Maximum Parsimony

Ulf Leser

Wissensmanagement in der
Bioinformatik

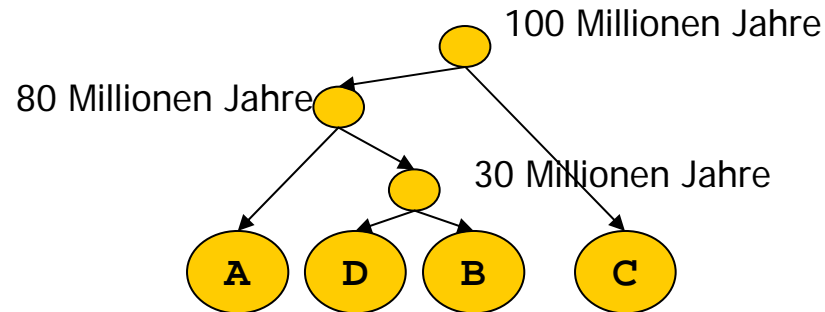


Voraussetzungen



- An innere Knoten eines Stammbaum kann man den Zeitpunkt der Speziation schreiben („branch points“)
- Wenn die Molecular Clock Theory gilt
 - Ist die Menge an Veränderungen auf einer Kante **proportional zu der verstrichenen Zeit**
 - Damit proportional zur Länge der Kante
 - Haben alle Geschwister den gleichen Abstand zum Elternknoten
 - Ist der Editabstand zweier Knoten (=Evolution dazwischen) proportional zur Summe der **Editabstände beider Knoten zum kleinsten gemeinsamen Vater**

Ultrametrien



- Wenn man den Baum und die Zeitpunkte weiß, dann gilt
 - Alle Zahlen auf einem Pfad von der Wurzel zu einem beliebigen Blatt nehmen strikt ab
 - Der **Zeitpunkt der Aufspaltung ist ein Abstandsmaß** für zwei Arten
 - Für Knoten X, Y sei $d(X, Y)$ das Label des kleinsten gemeinsamen Vorfahren
 - Im Beispiel: $d(A, B) = 80$, $d(B, C) = 100$, $d(A, D) = 80$
 - Das ist eine Metrik
 - $d(X, X) = 0$, $d(X, Y) > 0$, $d(X, Y) = d(Y, X)$, und $d(X, Y) \leq d(X, Z) + d(Z, Y)$
 - Es ist sogar eine **Ultrametrik** (gleich)

Ultrametrische Bäume

- Definition

*Sei T ein binärer, gewurzelter Baum und D eine symmetrische Matrix mit n Zeilen und n Spalten. T heißt **ultrametrischer Baum** für D wenn gilt:*

- *T hat n Blätter, beschriftet mit den Zeilen von D*
- *Jeder innere Knoten von T ist mit einem Wert aus D beschriftet*
- *Auf jedem Pfad von der Wurzel zu einem Blatt in T sind die Beschriftungen der inneren Knoten strikt abnehmend*
- *Für alle Blätter i, j mit $i \neq j$ gilt: der **kleinste gemeinsame Vorfahr** von i und j ist mit $D[i, j]$ beschriftet*

- Bemerkung

- Jeder Stammbaum ist ultrametrisch für die Abstandsmatrix mit den Speziationszeitpunkten als Abstandsmaß

Überlegungen

- Das kann auch nicht immer gehen
 - Matrix hat $(n^2-n)/2$ relevante Zellen
 - Baum hat nur $n-1$ innere Knoten
 - Eine Matrix, zu der man einen ultrametrischen Baum konstruieren kann, muss also Duplikate enthalten
- Definition

*Eine symmetrische Matrix D mit n Spalten und Zeilen ist **ultrametrisch**, wenn für beliebige Zeilen i, j, k gilt, dass das Maximum von $D[i,j]$, $D[j,k]$ und $D[i,k]$ genau zweimal vorkommt*
- Bemerkung
 - Also entweder
 - $D[i,j]=D[j,k]$ und $D[i,j]>D[i,k]$
 - $D[i,j]=D[i,k]$ und $D[i,j]>D[j,k]$
 - $D[j,k]=D[i,k]$ und $D[j,k]>D[i,j]$

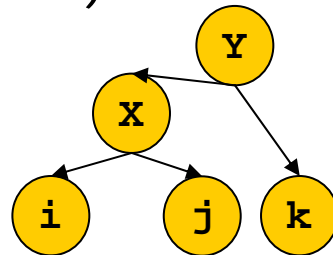
Von der Matrix zum Baum und zurück

- Theorem

Eine symmetrische Matrix D hat einen ultrametrischen Baum gdw. D ultrametrisch ist.

- Beweis

- (1) Nehmen wir erst an, dass zu D ein ultrametrischer Baum T existiert. Nehmen wir an, dass i, j, k wie folgt liegen (alle anderen Knoten können wir ignorieren)



- Dann gilt offensichtlich $D[i,k]=D[j,k]=Y$ und $D[i,k]>D[i,j]=X$
- Dito kann man für die zwei anderen möglichen Lagen von i, j, k
- Das gilt für alle Tripel von Knoten
- Also ist D ultrametrisch

Distanzbasierte Algorithmen

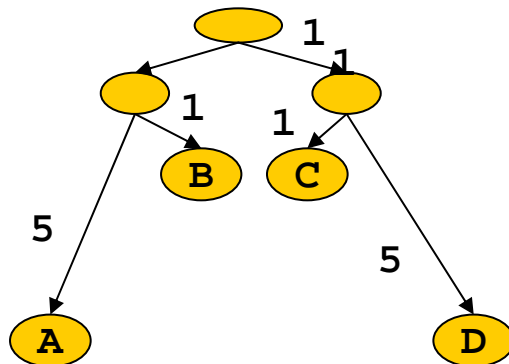
- Konstruktion des ultrametrischen Baumes basiert rein auf Distanzmaßen
 - Einen ultrametrischen Baum gibt es nicht für alle Matrizen
 - Es gibt weniger empfindliche Verfahren (gleich)
- Die Geschichte einzelner „Sites“ wird nicht berücksichtigt
- Solche Algorithmen zur Berechnung von Stammbäumen nennt man **distanzbasiert**
- Alternative: **Merkmalsbasierte Verfahren**
 - Beachten jedes einzelne Merkmal (Basen)
 - Vertreter: Perfect Phylogeny; Maximum Parsimony
 - Später

UPGMA - Hierarchisches Clustering

- UPGMA
 - „Unweighted pair group method with arithmetic mean“
 - Anderer Name: [Hierarchisches Clustering](#)
- Sehr einfaches und allgemeines Verfahren, kann bei allen möglichen Problemen angewandt werden
- [Wenn eine Matrix ultrametrisch ist](#), dann findet UPGMA den dazugehörigen ultrametrischen Baum
 - UPGMA nimmt also die Molecular Clock an – alle Pfade von einem Blatt zur Wurzel haben am Ende die selbe Länge
- Achtung: UPGMA konstruiert immer einen Baum
 - Auch wenn die Matrix nicht ultrametrisch ist

Ultrametrien und Sequenzabstände

- Realität
 - Matrix mit Editabständen der Sequenzen
- **Reale Daten** sind selten ultrametrisch
 - Spezies unterliegen spezifischem Selektionsdruck
- Die Zahl der Mutationen auf den Kanten zu Geschwistern sind nicht **gleich verteilt**
- Beispiel: Stammbaum, aber keine Ultrametrik

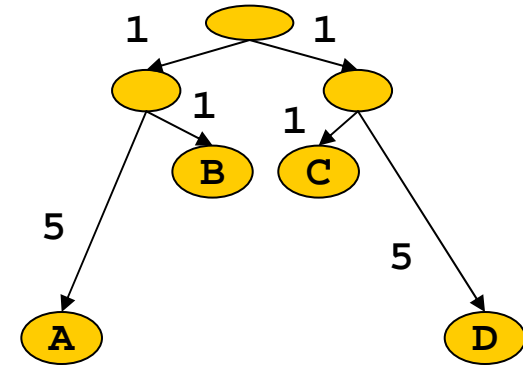


| | A | B | C | D |
|---|---|---|---|----|
| A | | 6 | 8 | 12 |
| B | | | 4 | 8 |
| C | | | | 6 |
| D | | | | |

Wo UPGMA irrt

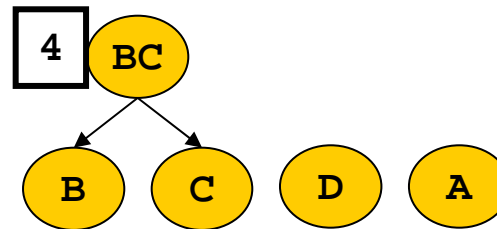
Der echte Baum

| | B | C | D |
|---|---|---|----|
| A | 6 | 8 | 12 |
| B | | 4 | 8 |
| C | | | 6 |
| D | | | |



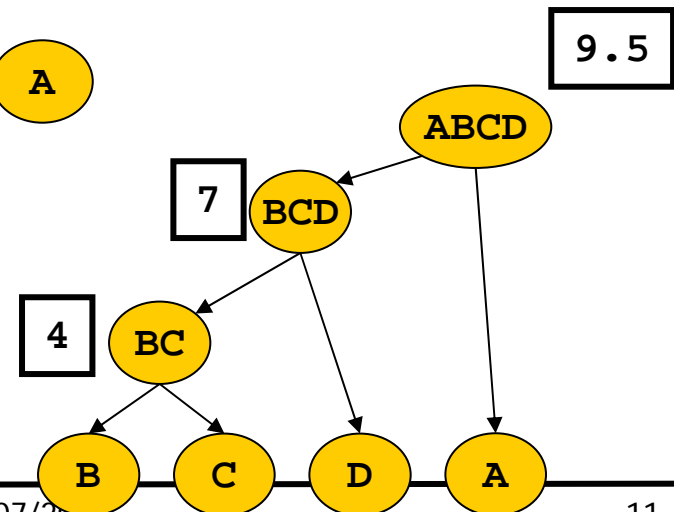
Was erzeugt UPGMA?

| | B | C | D |
|---|---|---|----|
| A | 6 | 8 | 12 |
| B | | 4 | 8 |
| C | | | 6 |



| | A | BC | D |
|----|---|----|----|
| A | | 7 | 12 |
| BC | | | 7 |

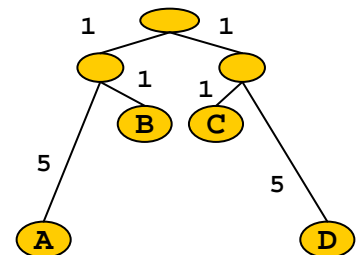
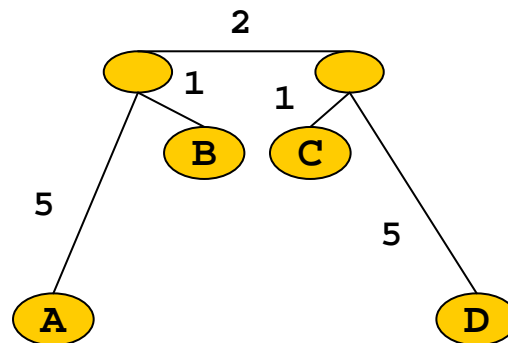
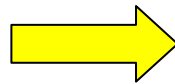
| | A |
|-----|-----|
| BCD | 9.5 |



Additive Bäume

- Ultrametrien: Label der inneren Knoten
- Andere Betrachtung: Berechnung der **Kantenlabel**
- Problem
Gegeben eine Ähnlichkeitsmatrix, finde einen binären Baum so, dass die **Summe der Kantenlabel** auf dem Pfad von jedem Knoten i zu jedem Knoten j gleich $D[i,j]$ ist
- Bemerkung
 - Da es nur Abstände gibt, hat der Baum zunächst **keine Wurzel**

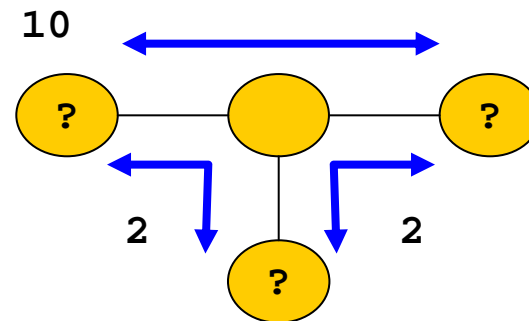
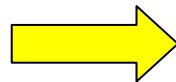
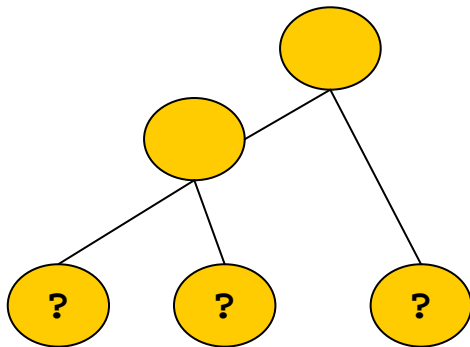
| | B | C | D |
|---|---|---|----|
| A | 6 | 8 | 12 |
| B | | 4 | 8 |
| C | | | 6 |
| D | | | |



Matrizen und additive Bäume

- Fragen
 - Existiert zu jeder Matrix ein additiver Baum?
 - Wie findet man einen additiven Baum zu einer gegebenen Matrix?
- Gegenbeispiel

| | A | B | C |
|---|---|----|---|
| A | | 10 | 2 |
| B | | | 2 |



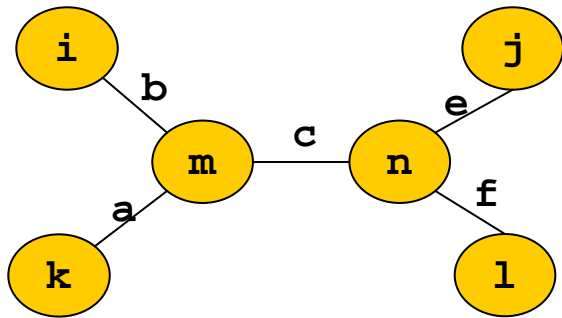
4-Punkt Bedingung

- Theorem

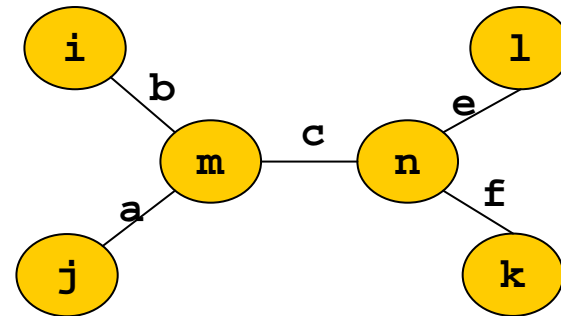
Eine Matrix D hat einen additiven Baum gdw. für alle Quadrupel i, j, k, l die 4-Punkt Bedingung gilt:

$$D(i,k) + D(j,l) \leq \max(D(i,j) + D(k,l), D(i,l) + D(k,j))$$

- Beweis: Literatur



$$(a+b) + (e+f) \leq \max((b+c+e) + (a+c+f), (b+c+f) + (a+c+e))$$



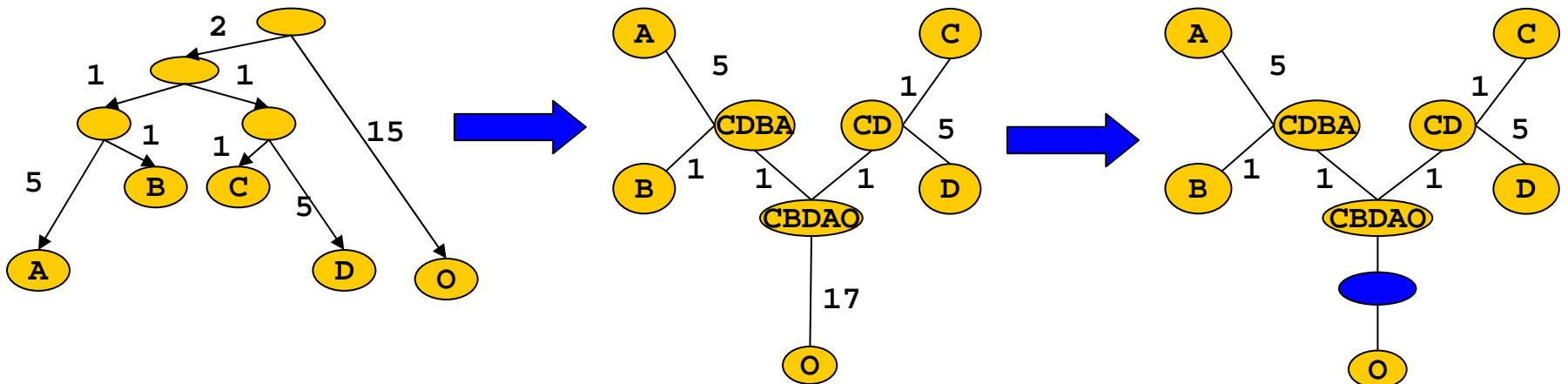
$$(b+c+f) + (a+c+e) \leq \max((a+b) + (e+f), (b+c+e) + a+c+f)$$

Neighbor-Joining

- Matrizen und Algorithmen
 - Ultrametrische Matrizen – UPGMA
 - **Additive Matrizen – Neighbor Joining**
- Hierarchisches Clusterverfahren (wie UPGMA)
 - Erzeugt einen binären Baum ohne Wurzel
 - **Grundaufbau wie UPGMA**
 - Beginne mit so vielen Clustern wie Blättern
 - Wähle nach **bestimmtem Kriterium zwei Cluster**
 - Verschmelze die zwei Cluster und verbinde Knoten im Baum
 - Iteriere, bis nur noch ein Cluster vorhanden ist
- Unterschiede
 - UPGMA wählt Cluster zur Verschmelzung nur nach Nähe zueinander
 - Neighbor Joining wählt Cluster **nach der Nähe zueinander und dem Abstand zu anderen Clustern**

Outgroups

- Eine **Outgroup** ist ein Taxa, das weiter von allen anderen Taxa entfernt ist also diese untereinander
 - Beispiel: Menschen, Mäuse, Ratten, Schweine – Stoch
- Was passiert mit der Outgroup?
 - NJ ordnet sie im Baum ein
 - Offensichtlich muss die Kante, die zu der Outgroup führt, den **Wurzelknoten** enthalten
 - Damit wird der ganze Baum zeitlich angeordnet



Wenn die Daten nun ...

- Weder ultrametrisch noch additiv sind?
- Approximieren; z.B. durch
 - Gegeben Abstandsmatrix D . Wähle eine Baumtopologie T und berechne die tatsächlichen Abstände im Baum $d(i,j)$ so, dass der Fehler minimiert wird

$$error(t) = \min_d \left(\sum_{i=1}^n \sum_{j \neq i} (D[i, j] - d(i, j))^2 \right)$$

- Für gegebene Topologie T ist das effizient lösbar, aber ...
- Man muss alle Topologien ausprobieren
- Damit ist das Problem NP-vollständig
- Oder man versucht **Heuristiken**: UPGMA, Neighbor Joining
- Oder ganz anders aufgebaute Methoden: Maximum Parsimony, Maximum Likelihood, ...

Inhalt dieser Vorlesung

- Characterbasierte Verfahren
- Perfect Phylogeny
- Maximum Parsimony
 - Fitch / Sankoff's Algorithmus
 - Heuristiken

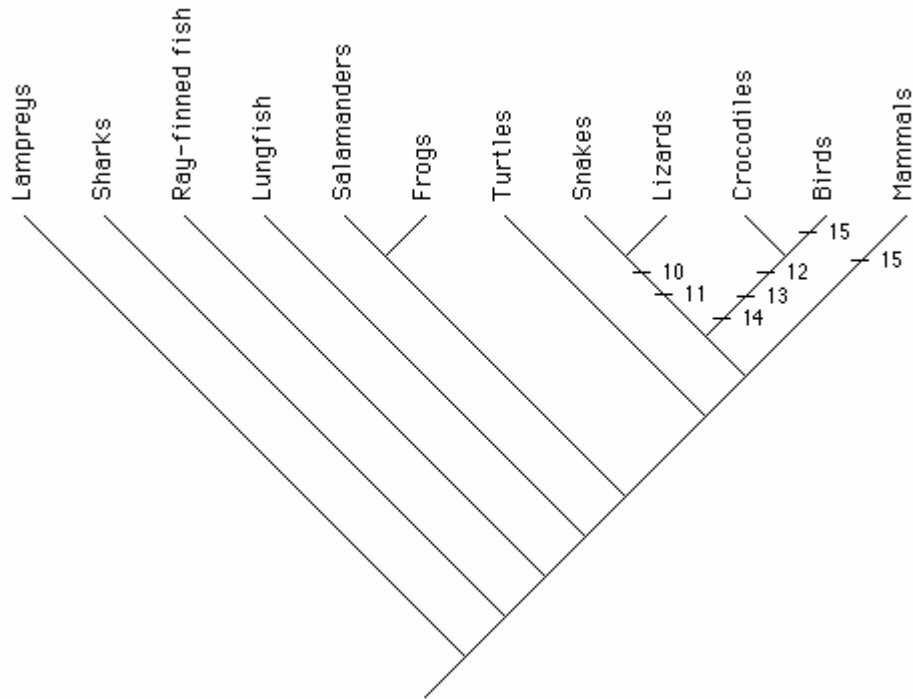
Distanz versus Zeichen

- Distanzbasierte Algorithmen abstrahieren von einzelnen Zeichen und basieren auf dem **Abstand von Taxa**
- **Characterbasierte** Verfahren betrachten die Entwicklung jedes einzelnen „Characters“
 - Nuklein- oder Aminosäure
 - Morphologische Eigenschaften
 - Vorhandensein / Abwesenheit bestimmter Gene/Funktionen
 - ...
- Character sollten in einem **Abstammungsverhältnis** stehen
 - Sequenzen müssen homolog sein
- Wahl der Character beeinflusst das Ergebnis erheblich
 - Eine „korrekte“ Wahl gibt es nicht – erheblicher Freiraum

| Character | Lampreys | Shar ks | Teleosts | Lungfis h | Frogs | Salama nders | Turtles | Lizards | Snakes | Crocodi les | Birds | Mamm als |
|-------------------------------|------------|------------------|---------------|---------------|------------------|------------------|------------------|------------------|------------------|------------------|----------|----------|
| 1 Internal skeleton | yes | yes | yes | yes | yes | yes | yes | yes | yes | yes | yes | yes |
| 2 Jaws | no | yes | yes | yes | yes | yes | yes | yes | yes | yes | yes | yes |
| 3 Ossified skeleton | no | no | yes | yes | yes | yes | yes | yes | yes | yes | yes | yes |
| 4 Internal nostrils | no | no | no | yes | yes | yes | yes | yes | yes | yes | yes | yes |
| 5 Atrial septum | no | no | no | yes | yes | yes | yes | yes | yes | yes | yes | yes |
| 6 Four limbs | no | no | no | no | yes | yes | yes | yes | yes | yes | yes | yes |
| 7 Teeth pedicillate | no | no | no | no | yes | yes | no | no | no | no | no | no |
| 8 Amniotic egg | no | no | no | no | no | no | yes | yes | yes | yes | yes | yes |
| 9 Temporal fenestrae | none | none | none | none | none | none | none | two | two | two | two | one |
| 10 Hemipenes | no | no | no | no | no | no | no | yes | yes | no | no | no |
| 11 Suspensorium streptosylous | no | no | no | no | no | no | no | yes | yes | no | no | no |
| 12 Antorbital fenestrae | no | no | no | no | no | no | no | no | no | yes | yes | no |
| 13 Lateral fenestrae ossified | no | no | no | no | no | no | no | no | no | yes | yes | no |
| 14 Gizzard | no | no | no | no | no | no | no | no | no | yes | yes | no |
| 15 Homeothermy | no | no | no | no | no | no | no | no | no | no | yes | yes |
| 16 Body covering | scale-less | dermal denticles | dermal scales | dermal scales | smooth epidermis | smooth epidermis | epidermal scales | epidermal scales | epidermal scales | epidermal scales | feathers | hair |

Quelle: Morrison, Phylogenetic-Tree Building, 1996

Abgeleiteter Baum



Gesucht: Der Baum mit den **wenigsten Änderungen**

Maximum Parsimony

- Generelles Prinzip
 - Findet man in vielen Problemen / Disziplinen
 - „Occam's razor“ (William of Ockham, UK, 14 Jhdt.)
 - *„One should not increase, beyond what is necessary, the number of entities required to explain anything“*
 - KISS principle – „Keep it as simple as possible“
- Maximum Parsimony in der Phylogeny
 - Zustände der Character in verschiedenen Spezies mit **so wenig evolutionären Ereignissen wie möglich** erklären
- Wahl eines Baumes „objektiveren“
 - Denn es gibt unendlich viele Bäume (0-1-0-1-0-1-0-....)

MP und Multiple Sequence Alignment

- Verwendet man Sequenzen, ist jeder Buchstabe (Nukleinsäure, Aminosäure) ein Character
- Welche Basen muss man dabei miteinander vergleichen?
- Vorgehen
 - Multiple Sequence Alignment berechnen
 - Spalten mit Insdels werden i.d.R. ignoriert
 - Oft: Löschen der Bereiche großer Variabilität
- Aber – MSA benötigt oft einen evolutionären Baum
 - Also: Iterieren

| taxon |10..... |20..... |30..... |40..... |50 |
|-----------------|------------------------|-------------------|------------------------|---------------|---------|
| Fu Nosema.40928 | QFGLFSPEEIRASSVALIR-- | YPETLENG-- | VFKESGLVCAGHFGHIELVK | | |
| Fu Aspergillus. | QFGLFSPEEIKRMSVHVVE-- | YPETMDEQRQRERTKGL | ECPGHFGHIELAT | | |
| Ap Plasmodium.3 | ELGVLDPEI IKKISVCEIV-- | NVDIYKDG-- | FRREGGLYCPGHFGHIELAK | | |
| An Cricetulus.2 | QFGLVSPDELKRMSVT | EGGIKYPETTE-- | GGRKLGGLCECPGHFGHIELAK | | |
| An Homo.7434727 | QFGLVSPDELKRMSVT | EGGIKYPETTE-- | GGRKLGGLCECPGHFGHIELAK | | |
| An Drosophila.9 | QFGILSPDEIRMSVT | EGGVQFAETME-- | GGRKLGGLCECPGHFGHIDLAK | | |
| An Celegans.133 | QFGILSPDEIKRMSVAH-- | VEFPPEVYE-- | NGKRLGGLDCPGHFGHLELAK | | |
| Fu Spombe.54881 | QFGILSPDEIRMSVAK-- | IEFPETMDESGQR | RVGGLDCECPGHFGHIELAK | | |
| Pl Athaliana.40 | QFGILSPDEIRQMSVH--- | VEHSETTEKKGK | KVGGLECPGHFGYLELAK | | |
| My Ddiscoideum. | ----- | ----- | ----- | ECPGHFGHIELAK | |
| Rh Porphyra.316 | ----- | ----- | ----- | ECPGHFGHIELAK | |
| Kt Thrucei.1021 | QFEIFKERQIKSYAVCL | VEHAKSYANA---- | AIQSGEABCPGHFGYIELAE | | |
| Kt Leishmania.7 | QFEVFKEAQIKAYAKCI | IEHAKSYEHG---- | QFVRGGIECPGHFGYVELAE | | |

-
- Bevor wir das allgemeine MP Problem lösen ...

- **Perfect Phylogeny**

- Stark eingeschränktes Modell
- Alle Character sind **binär**: vorhanden oder nicht
- Wurzel hatte keine Eigenschaften
- Jede **Eigenschaft darf im Baum nur genau einmal erzeugt** werden
- Für Sequenzen unrealistisch, für komplexe Eigenschaften nicht
 - Und: führt zu eleganten Algorithmen

Perfect Phylogeny

- Definition

*Sei D eine binäre Matrix aus n Zeilen (Arten) und m Spalten (Character) mit $D[i,j]=1$ gdw Art i Eigenschaft j hat. T heißt **perfekter phylogenetischer Baum** für D gdw*

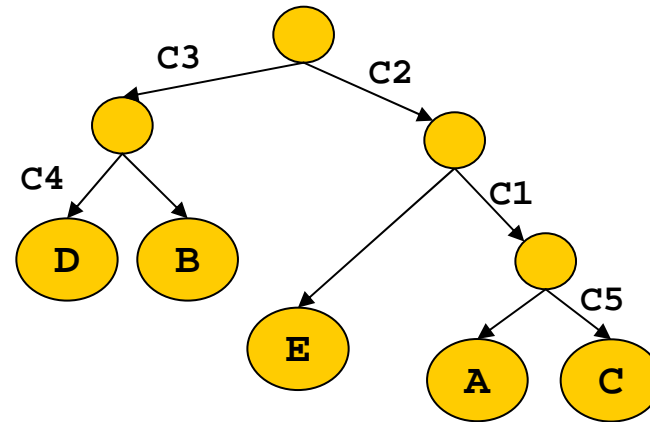
- *T hat n Blätter, beschriftet mit den Zeilen von D*
- *Jeder Character, der in mindestens einer Art vorhanden ist, steht an genau einer Kante von T*
- *Für jede Art i gilt, dass die Beschriftungen der Kanten auf dem Pfad von der Wurzel zu i genau die Character sind, die i besitzt*

- Bemerkungen

- Nicht an jeder Kante von T muss ein Character stehen, aber jeder Character muss an **genau einer Kante** stehen
- Character werden auf ihrer (und sonst nirgends) Kante „angeschaltet“ und dann nie mehr „abgeschaltet“

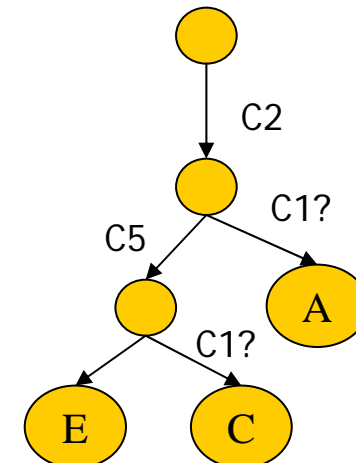
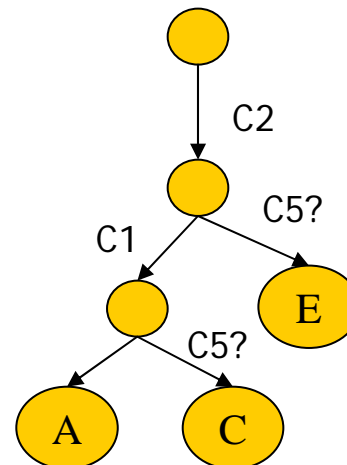
Beispiel

| | C1 | C2 | C3 | C4 | C5 |
|---|----|----|----|----|----|
| A | 1 | 1 | 0 | 0 | 0 |
| B | 0 | 0 | 1 | 0 | 0 |
| C | 1 | 1 | 0 | 0 | 1 |
| D | 0 | 0 | 1 | 1 | 0 |
| E | 0 | 1 | 0 | 0 | 0 |



Klappt das immer?

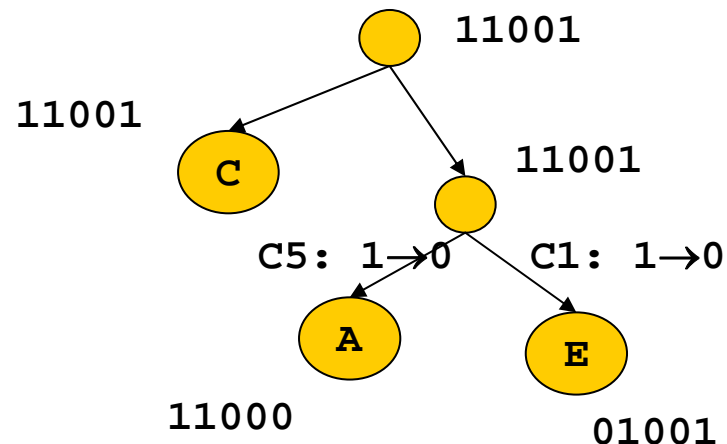
| | C1 | C2 | C3 | C4 | C5 |
|---|----|----|----|----|----|
| A | 1 | 1 | 0 | 0 | 0 |
| C | 1 | 1 | 0 | 0 | 1 |
| E | 0 | 1 | 0 | 0 | 1 |



Einschränkungen aufheben

- Zu jeder binären Matrix gibt es einen phylogenetischen Baum, wenn...
 - ein Character an mehreren Stellen wechseln darf („convergent evolution“)
 - Character an der Wurzel nicht abwesend sein müssen und später auch verschwinden dürfen

| | C1 | C2 | C3 | C4 | C5 |
|---|----|----|----|----|----|
| A | 1 | 1 | 0 | 0 | 0 |
| C | 1 | 1 | 0 | 0 | 1 |
| E | 0 | 1 | 0 | 0 | 1 |



Existenz perfekter phylogenetischer Bäume

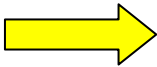
- Vorarbeiten

- Sei D eine binäre $n \times m$ Matrix und i eine Spalte (=Character)
- Wir erzeugen die **sortierte Matrix D'** aus D wie folgt
 - Interpretiere jede Spalte als **binäre Zahl mit der signifikantesten Stelle** in Zeile 1
 - Sortiere die Spalten in D absteigend nach diesen Zahlen
- Mit $Z(i)$ bezeichnen wir die Menge aller Arten (Zeilen), die die Eigenschaft i besitzen (also in i eine 1 haben)

- Bemerkung

- Das ändert nichts an unserem Problem, da alle Character unabhängig voneinander und in der Matrix beliebig angeordnet sind

| | C1 | C2 | C3 | C4 | C5 |
|---|----|----|----|----|----|
| A | 1 | 1 | 0 | 0 | 0 |
| B | 0 | 0 | 1 | 0 | 0 |
| C | 1 | 1 | 0 | 0 | 1 |
| D | 0 | 0 | 1 | 1 | 0 |
| E | 0 | 1 | 0 | 0 | 0 |
| | 20 | 21 | 11 | 2 | 4 |



| | C2 | C1 | C3 | C5 | C4 |
|---|----|----|----|----|----|
| A | 1 | 1 | 0 | 0 | 0 |
| B | 0 | 0 | 1 | 0 | 0 |
| C | 1 | 1 | 0 | 1 | 0 |
| D | 0 | 0 | 1 | 0 | 1 |
| E | 1 | 0 | 0 | 0 | 0 |
| | 21 | 20 | 11 | 4 | 2 |

Theorem und Beweis

- Theorem

*Sei D eine binäre Matrix. Die sortierte Matrix D' hat einen **perfekten phylogenetischen Baum** T gdw. für jedes Paar i, j von Spalten gilt:*

$$Z(i) \cap Z(j) = \emptyset \text{ oder } Z(i) \subseteq Z(j) \text{ oder } Z(j) \subseteq Z(i)$$

- Beweis

- Richtung „ \Rightarrow “: Sei T ein perfekter phylogenetischer Baum für D' , i, j zwei Spalten, und e_i bzw. e_j die Kanten in T , die mit i bzw. j beschriftet sind
 - Damit: Die Menge der Blätter unter e_i bzw. e_j ist exakt $Z(i)$ bzw. $Z(j)$
 - Folgende Fälle können in T auftreten
 - e_j liegt auf dem Pfad von der Wurzel zu e_i . Dann gilt $Z(j) \subseteq Z(i)$
 - e_i liegt auf dem Pfad von der Wurzel zu e_j . Dann gilt $Z(i) \subseteq Z(j)$
 - e_j und e_i liegen in unterschiedlichen Pfaden. Dann gilt $Z(i) \cap Z(j) = \emptyset$

Gegenrichtung

- Richtung „ \Leftarrow “: Sei D' wie verlangt. Wir konstruieren den perfekten phylogenetischen Baum T zu D' (und zeigen dabei auch die Eindeutigkeit des Baumes)
 - Seien p und q zwei Zeilen und k die **rechteste Spalte** mit $D'[q,k]=D'[p,k]=1$.

| | ... | i | ... | k | $k+1$ | ... |
|-----|-----|-----|-----|-----|-------|-----|
| ... | ... | ? | ? | ? | ? | ... |
| p | ... | ? | ? | 1 | ? | ... |
| ... | ... | ? | ? | ? | ? | ... |
| q | ... | ? | ? | 1 | ? | ... |
| ... | ... | ... | ... | ... | ... | ... |

Nicht beide 1

Gegenrichtung

- Richtung „ \leftarrow “
 - Wir zeigen erst: Für jede Spalte i links von k gilt $D'[p,i]=D'[q,i]$
 - Sei i eine Spalte links von k mit $D'[p,i]=1$. Gemäß Voraussetzung gibt es für $Z(k)$ und $Z(i)$ drei Möglichkeiten:
 - $Z(k) \cap Z(i) = \emptyset$; kann nicht sein, weil $p \in Z(k) \cap Z(i)$
 - $Z(k) \subseteq Z(i)$; dann muss $D'[q,i]=1$ gelten, weil $D'[q,k]=1$
 - $Z(i) \subseteq Z(k)$; dann muss $Z(i)=Z(k)$ sein, sonst kann i nicht links von k liegen
 - Weil p und q vertauschbar sind, folgt, dass $D'[p,i]$ und $D'[q,i]$ für alle i links von k entweder beide 0 oder beide 1 sind

| | ... | i | ... | k | $k+1$ | ... |
|-----|-----|-----|-----|-----|-------|-----|
| ... | ... | ? | ? | ? | ? | ... |
| p | ... | 1 | 0 | 1 | ? | ... |
| ... | ... | ? | ? | ? | ? | ... |
| q | ... | 1 | 0 | 1 | ? | ... |
| ... | | ... | ... | ... | ... | ... |

Gegenrichtung 2

- Wir haben bisher
 - $\forall i \leq k: D'[p,i]=D'[q,i]$
 - $\forall j > k: \text{Entweder } D'[p,j]=D'[q,j]=0 \text{ oder } D'[p,j] \neq D'[q,j]$
- Sei q_s die Zeile von q in D' **interpretiert als String** plus ein „\$“
 - Für alle Paare p, q gilt, dass q_s und p_s identisch sind bis zu einer bestimmten Position (k) und danach niemals an derselben Stelle eine 1 haben
 - Wegen „\$“ kann kein String einer Zeile Präfix eines anderen sein
- Wir bauen einen **Keyword Tree** T für alle Strings von Zeilen von D'
- T ist der **perfekte phylogenetische Baum** für D' (nach Löschen aller „\$“)
 - Für alle Paare p, q ist der Pfad in T bis zu einem Knoten nach k identisch, danach können nur noch verschiedene Character im p bzw. q Ast erscheinen (keine gemeinsame 1 nach k)
 - Keine Eigenschaft vor k kann irgendwo sonst im Baum als Kantenbeschriftung auftreten
- qed.

Komplexität

- Komplexität des **Existenztests** eines perfekten phylogenetischen Baums zu D ?
 - Wir haben $O(m^2)$ Spaltenvergleiche
 - Und müssen jeweils $O(n)$ Zeilen vergleichen (Enthaltensein)
 - Also $O(nm^2)$
- **Konstruktion** des perfekten phylogenetischen Baum, wenn er existiert
 - Konstruktion des Keyword-Trees ist $O(mn)$
 - Aber wir müssen erst sicherstellen, dass es einen phylogenetischen Baum gibt
- Es gibt auch $O(mn)$ Algorithmen zur gleichzeitigen Prüfung der Matrix und Konstruktion des Baums
 - Siehe Gusfield

Abschwächungen der Voraussetzungen

- Generalized perfect phylogeny
 - Jeder Character darf z Zustände haben
 - Jeder Zustand darf nur **einmal im Baum angenommen** werden (nach maximal $z-1$ vorherigen Wechseln)
 - Problem ist **NP-vollständig** (falls z beliebig)
- Aber wir gehen gleich zum allgemeinen Problem

-
- Maximum parsimony
 - Definition
 - Small parsimony: Fitch's Algorithmus
 - Large parsimony: Branch & Bound

Phylogenetische Bäume

- Definition

Gegeben eine Matrix D mit n Arten und m Character.

- *Character können Werte aus einer Menge Z mit $|Z|=z$ annehmen.*
- *Für alle i, j gilt, dass $0 < D[i, j] \leq z$.*

*Ein Baum T heißt **phylogenetischer Baum** für D wenn gilt:*

- *T ist ein binärer Baum mit n Blättern, beschriftet mit den Zeilen von D*
- *Jeder innere Knoten k (inklusive der Wurzel) ist beschriftet mit einem Label aus einem Zustand pro Character: $\text{label}(k) \in Z^m$*

- Bemerkungen

- Erweiterung auf unterschiedliche Zustandsmengen pro Character ist trivial

- Es existieren **sehr viele phylogenetische Bäume** zu einer Matrix D

- Zahl binäre Baumtopologien $\frac{(2n-3)!}{2^{n-2} * (n-2)!}$
- Jeder hat $n-1$ innere Knoten
- Jeder innere Knoten kann z^m verschiedene Label haben

Maximum Parsimony

- Definition

- Sei T ein phylogenetischer Baum zu einer Matrix D . T habe die Kantenmenge E . Der *parsimony score* $S(T)$ von T ist definiert als:

$$S(T) = \sum_{(u,v) \in E} |\{j \mid v_j \neq u_j\}|$$

- Ein phylogenetischer Baum T für eine Matrix D heißt *maximal parsimony*, wenn $S(T)$ der kleinstmögliche *parsimony score* aller phylogenetischen Bäume von D ist.

- Bemerkungen

- u_j ist der Zustand des Characters j in Knoten u

Wie finden wir den?

- Wir zerlegen das Problem
 - „Small parsimony“: Feste Baumtopologie
 - „Large parsimony“: Beliebige Topologie

Small Parsimony

- Definition Small Parsimony Problem
Geg. eine feste Baumtopologie T und eine Taxa/Character Matrix D . Das Small Parsimony Problem (SPP) sucht nach den Labels der inneren Knoten so, dass der Parsimony Score von T minimal ist.
- Beobachtung
 - Alle Character sind **unabhängig voneinander**
 - Diese Annahme steckt in dem gesamten MP Ansatz
- Damit kann man das SPP wie folgt lösen
 - Berechne die **besten Label pro Character**
 - Setze die Knotenlabel aus den einzelnen Characterlabeln zusammen
- Also reduziert sich das $n \cdot m$ Problem auf **$m \cdot n \cdot 1$ Probleme**
 - m Probleme mit jeweils einer Matrix mit genau einem Character C

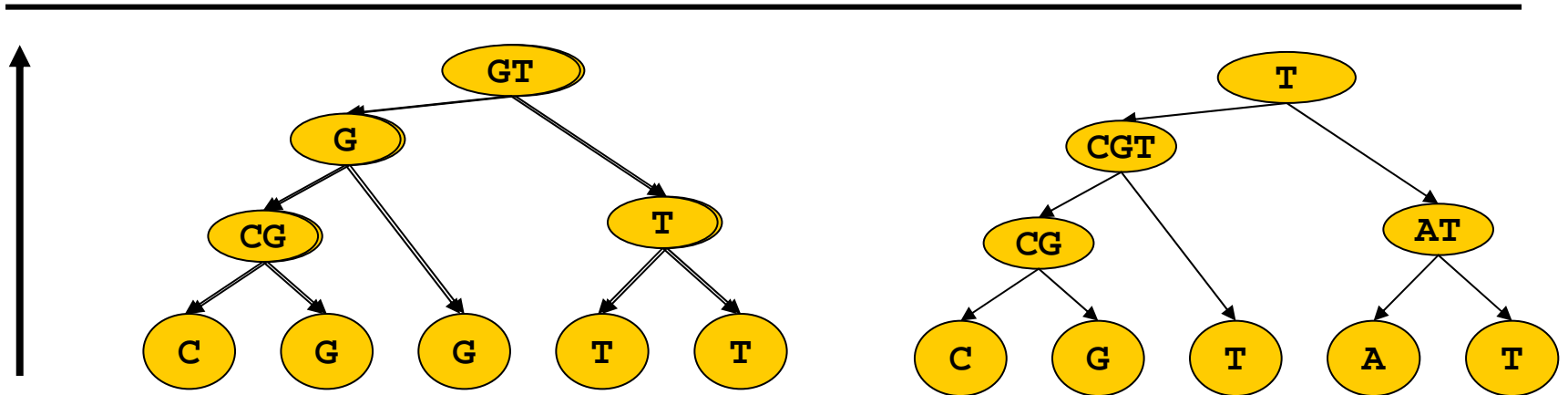
Fitch's Algorithmus

- Zwei Phasen
 - Wir berechnen bottom-up **mögliche Label P** pro Knoten
 - Dann legen wir top-down die **Label pro Knoten fest**
- Phase 1: Berechne $P(k)$ für alle Knoten k von T
 - Wenn k ein Blatt ist, dann $P(k)=k_c$
 - Sonst habe k Kinder u und v . Dann

$$P(k) = \begin{cases} P(u) \cap P(v), & \text{falls } P(u) \cap P(v) \neq \emptyset \\ P(u) \cup P(v) & \text{sonst} \end{cases}$$

- Bemerkung
 - Wir berechnen die P bottom-up
 - Intuitiv: Wenn es eine **Gemeinsamkeit gibt**, dann nutze sie aus und propagiere nur diese nach oben; sonst nimm alle Möglichkeiten

Beispiel

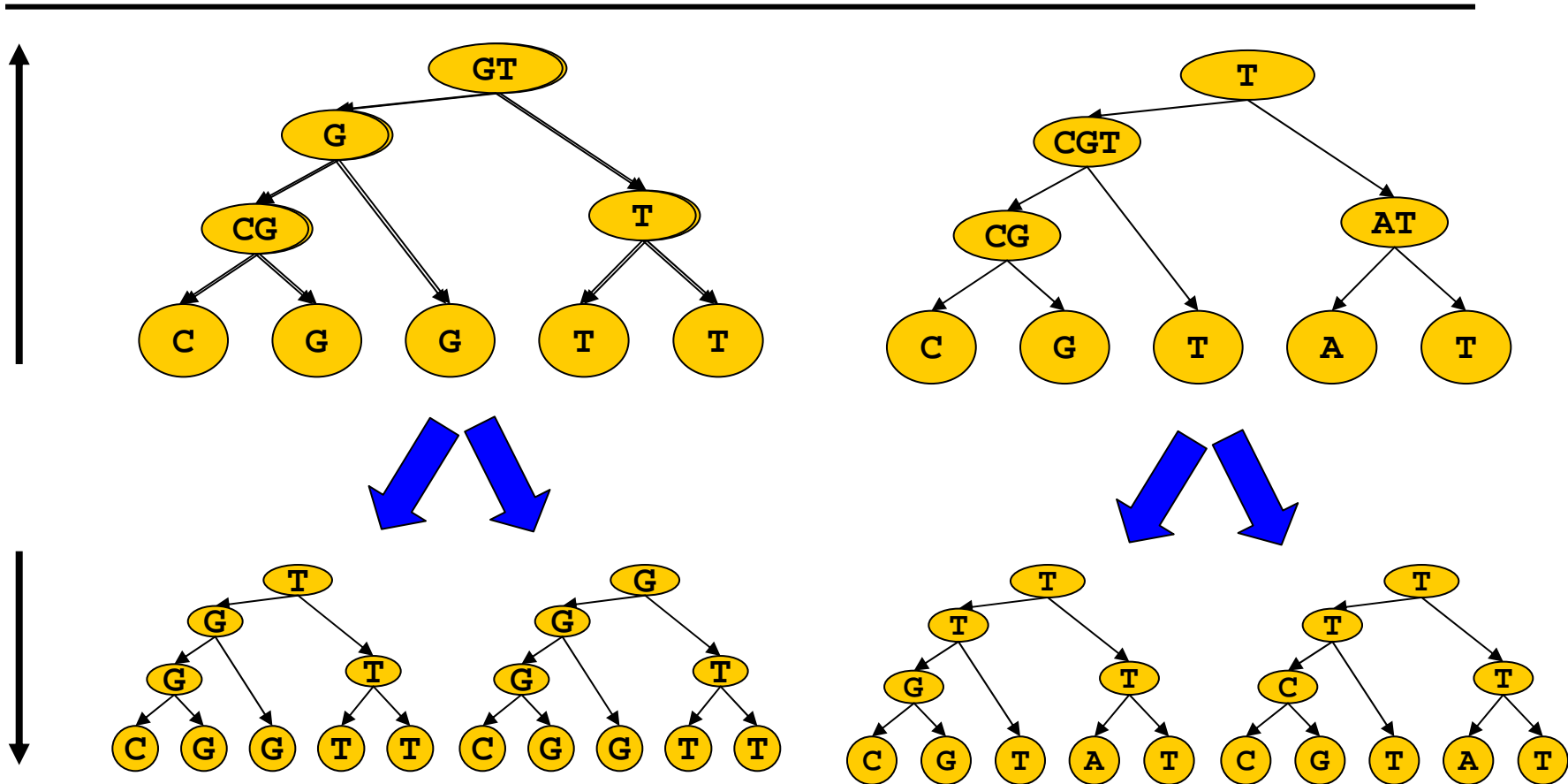


Fitch, Phase 2

- Phase 2
 - Wähle $\text{label}(\text{root})$ beliebig aus $P(\text{root})$
 - **Traversiere** alle inneren Knoten k
 - Wenn $\text{label}(\text{parent}(k)) \in P(k)$, dann setze $\text{label}(k) = \text{label}(\text{parent}(k))$
 - Sonst wähle $\text{label}(k)$ beliebig aus $P(k)$
- Theorem

*Fitch's Algorithmus berechnet ein Labeling von T mit dem **kleinstmöglichen Parsimony Score**.*
- Beweis
 - Literatur

Beispiel

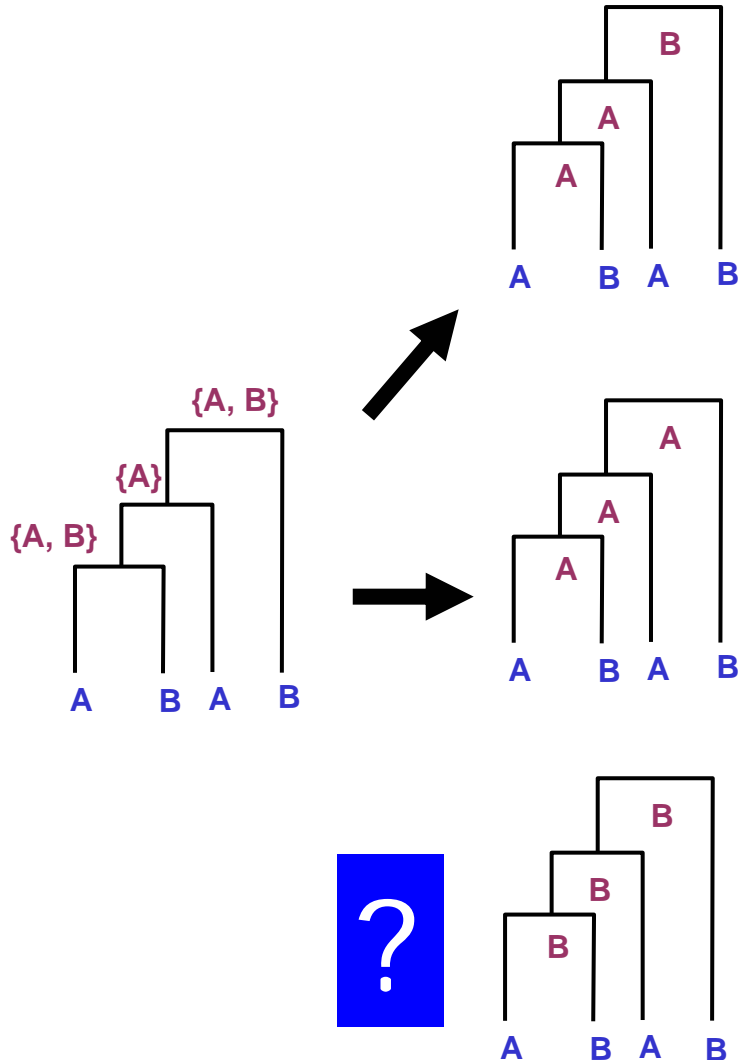


Scores: **2** **2** **3** **3**

Komplexität von Fitch's Algorithmus

- Komplexität
 - Beachte: Jedes P kann maximal z Elemente haben
 - Phase 1: Für jeden inneren Knoten müssen wir $O(z)$ Vergleiche machen, um P auszurechnen – also $O(n * z)$
 - Phase 2: Traversierung aller innerer Knoten ist $O(n)$; dazu ein $O(\ln(z))$ Test auf Enthaltensein des Vaterlabels im Label jedes Kindes
 - Zusammen: $O(n * z + n * \ln(z)) = O(n * z)$

Aber Vorsicht



- Fitch's Algorithmus findet **nicht alle optimalen Bäume**
 - Algorithmus ist greedy
 - Erkennt nicht, dass einen Wechsel in Kauf zu nehmen sich später auszahlen kann
- Verbesserung (und Verallgemeinerung)
 - **Weighted Parsimony**
 - **Sankoff's Algorithmus**

Weighted Parsimony

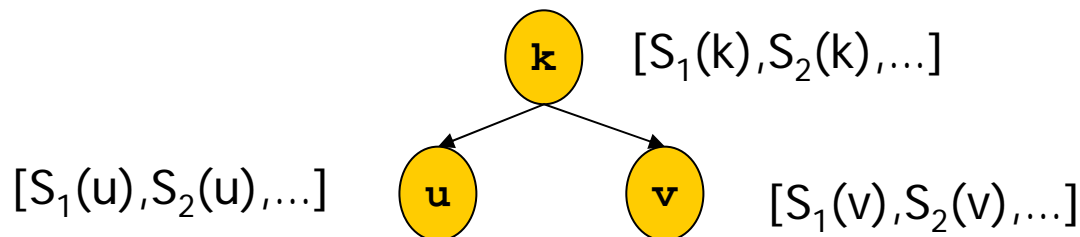
- Bisher nehmen wir an, dass alle Änderungen von Zuständen eines Characters gleich viel kosten (nämlich 1)
 - Schlecht: Siehe PAM, BLOSSUM, etc.
- Nahe liegende Erweiterung: **Weighted Parsimony**
 - Kosten der Zustandsübergänge werden mit Hilfe einer Substitutionsmatrix S gewichtet
- Problemformulierung
 - Gegeben eine Baumtopologie T , eine Substitutionsmatrix S und eine Matrix D
 - Finde eine Beschriftung der inneren Knoten von T so, dass der folgende Ausdruck minimiert wird

$$S^w(T) = \sum_{(u,v) \in E(T)} S(\text{label}(v), \text{label}(u))$$

Sankoff's Algorithmus

- Wieder zwei Phasen
 - Berechne für **jeden Knoten k** und **jeden Zustand z** die minimalen Kosten $S_z(k)$ des Baumes unter k, wenn k mit z beschriftet wird
 - Zweite Phase legt dann die Label fest
- Phase 1
 - Für alle Blätter, setze $S_z(k) = \begin{cases} 0, & \text{wenn } label(k) = z \\ \infty & \text{sonst} \end{cases}$
 - Traversiere den Baum bottom-up und berechne für jeden Knoten k mit Kindern u und v (z läuft über alle Zustände):

$$S_z(k) = \min_i (S(z, i) + S_i(u)) + \min_i (S(z, i) + S_i(v))$$



Sankoff's Algorithmus

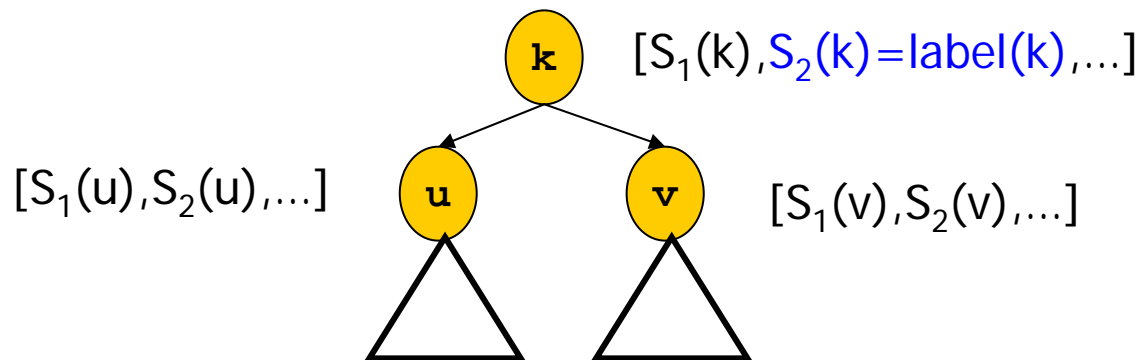
- Phase 2 (Top-down)

- Wurzel:

$$label(root) = \min_i(S_i(root))$$

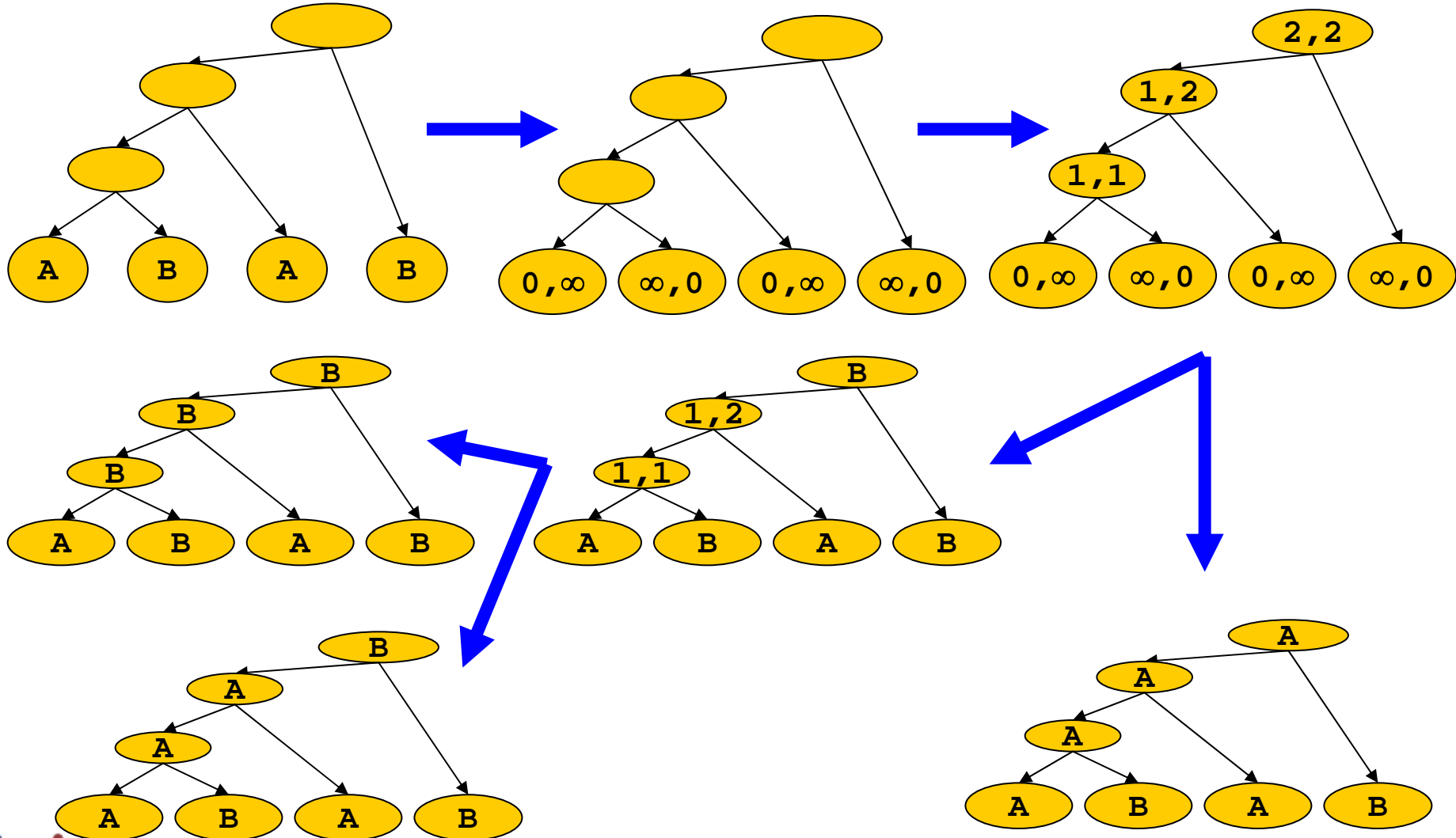
- Knoten u mit Vater k:

$$label(u) = \min_i(S(label(k), i) + S_i(u))$$



Beispiel – Was bei Fitch schief ging

| | | |
|---|---|---|
| | A | B |
| A | 0 | 1 |
| B | 1 | 0 |

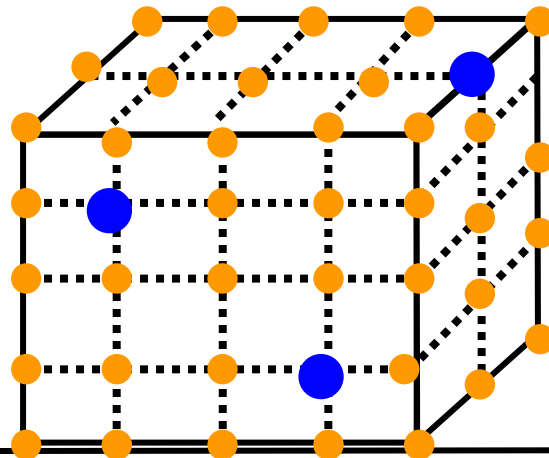


Large parsimony

- Small parsimony kann man also effizient lösen
- Definition Problem Large Parsimony
*Geg. eine Taxa/Character Matrix D . Das **Large Parsimony Problem (LPP)** sucht nach der Baumtologie T und den Labels der inneren Knoten so, dass der Parsimony Score von T minimal ist.*
- Unglücklicherweise gilt
 - LPP ist NP-vollständig
 - Im Prinzip müssen wir also alle möglichen Topologien ausprobieren
 - Warum ist das so (Sketch)?

MP und Steiner Bäume

- Reduktion auf **Steiner-Baum Problem** auf m-dimensionalen Hypercube
 - ... also kann man die Lösung leicht bis auf Faktor 2 approximieren
- Gegeben m Character mit jeweils z Zustände
 - Das spannt einen m-dimensionalen Raum auf
 - Bilde den Graphen G mit Knoten für alle Gitterpunkte und Kanten zwischen Knoten entlang aller Achsen
 - Jedes Taxa ist ein Punkt in diesem Raum / Knoten im Graph
 - Der Abstand zweier Taxa ist die Zahl ihrer Koordinaten mit ungleichen Werten
 - Das MP Problem ist jetzt äquivalent zu: Finde den Steiner Baum für alle Taxa-Punkte in G



Branch & Bound

- Heuristik zur Lösung: **Branch & Bound**
- Beobachtung
 - Der Parsimony Score eines Baumes wird durch Hinzufügen eines neuen Blattes **niemals kleiner**

Branch & Bound Algorithmus

- Gegeben eine Matrix D
- Rekursive Tiefensuche durch alle möglichen Topologien. Berechne optimales $S(T)$ für jeden (wachsenden) Baum
 - Beginne mit allen Topologien für die ersten 3 Arten
 - Zähle alle Möglichkeiten auf, die 4. Art hinzuzufügen
 - Bei k bisherigen Arten im Baum, gibt es $2^k - 1$ Möglichkeiten
 - Halte eine davon fest und steige weiter ab (5. Art, 6. Art ...)
- Am Blatt des Suchbaums haben wir eine komplette Topologie K für D mit optimalem Score $S = S(K)$
 - Der ist natürlich i.A. nicht optimal für D
- Traversiere den Rest des Baums
 - Immer, wenn ein (Teil-)Baum einen Score größer S hat, vergiss diesen Ast des Suchraums (Pruning)
 - Passe S ständig an das aktuelle Optimum an

Eigenschaften

- Vergleichbar dem A^* Algorithmus
- **Worst-Case** ist immer noch exponentiell
 - Aber „normale“ Laufzeiten sind deutlich besser
- Idee zum schnellen Finden einer i.A: guten oberen Schranke
 - Topologie durch Neighbor Joining bestimmen
 - Bestes Labeling mit Fitch/Sankoff berechnen
 - Dessen Score als obere Schranke benutzen und B&B starten
- Viele weitere **Heuristiken** möglich/nötig
 - Mit welchen Arten soll man beginnen?
 - Welche Bäume soll man zuerst aufzählen?
 - Vielleicht immer die besten 2 Kinder weiterverfolgen?
 - ...

Andere Möglichkeiten

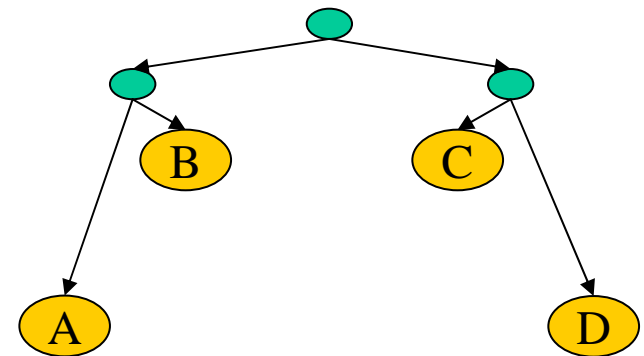
- **Iterative Verbesserung**
 - Beginne mit irgendeiner Topologie und berechne optimalen Score
 - Verändere diese „lokal“, nach Möglichkeit zum Guten
 - So lange, bis es nicht mehr besser wird
 - Wiederhole das mit vielen zufälligen Startbäumen
- **Greedy**
 - Zähle alle Bäume mit wachsender Größe auf
 - Berechne für jede Topologie jeweils den besten Score
 - Wähle jeweils den besten Baum, und erweitere nur diesen
- Viele weitere Heuristiken bekannt
- Keine Garantie für Finden des globalen Optimum

Weitere Beobachtung

- **Uninformative characters:** Wir können aus der Matrix alle Spalten löschen, für die gilt
 - Alle Werte sind gleich, oder
 - Jeder Wert kommt nur einmal vor (das kostet z Änderungen, egal wie der Baum aussieht), oder
 - Es gibt keine zwei Zustände, die mindestens zweimal vorkommen (z.B.: `XXYZ`) – kostet immer 2
- Dadurch wird das Small Parsimony Problem billiger
- Aber die Komplexität des Large Parsimony Problems ist leider in n (Arten), nicht in m (Character)

„Felsenstein Zone“

- Eine Methode ist "**statistisch konsistent**", wenn die Wahrscheinlichkeit, dass sie bei gegebenen Daten den richtigen Baum ausrechnet, mit wachsender Länge der Eingabe (also Länge der Sequenzen) gegen 1 geht
- MP ist nicht statistisch konsistent
 - Siehe Beispiel
 - Je länger die Kanten zu A und D sind ...
 - Desto größer die Unterschiede zwischen A,B und D,C
 - Desto größer die Wahrscheinlichkeit, dass A und D durch Mehrfachmutationen zufällig ähnlich werden
 - „**Long branch attraction**“ – A und D werden im Baum als Nachbarn eingeordnet
- Ursache
 - MP normiert nicht über die Sequenzlänge
 - MP ignoriert (wie alle Methoden bisher) Mehrfachmutationen



Vergleich

- Man liest häufiger, dass alle Phylogeniemethoden recht gut funktionieren
 - Gilt nur bei einfachen **Evolutionsmodellen**
 - Güte hängt von den Eigenschaften der Daten ab
- Distanzbasierte Methoden
 - Am ungenauesten, dafür schnell
 - Brauchen numerische Abstandsmasse
- Maximum Parsimony: Besser, aber sehr teuer
- **Maximum Likelihood**: Noch besser, noch teurer
- Also: Mehrere Methoden vergleichen
 - Gruppen, die überall gleich sind, gelten als sehr robust
 - „**Consensus tree**“