Bioinformatik

Character-basierte Verfahren

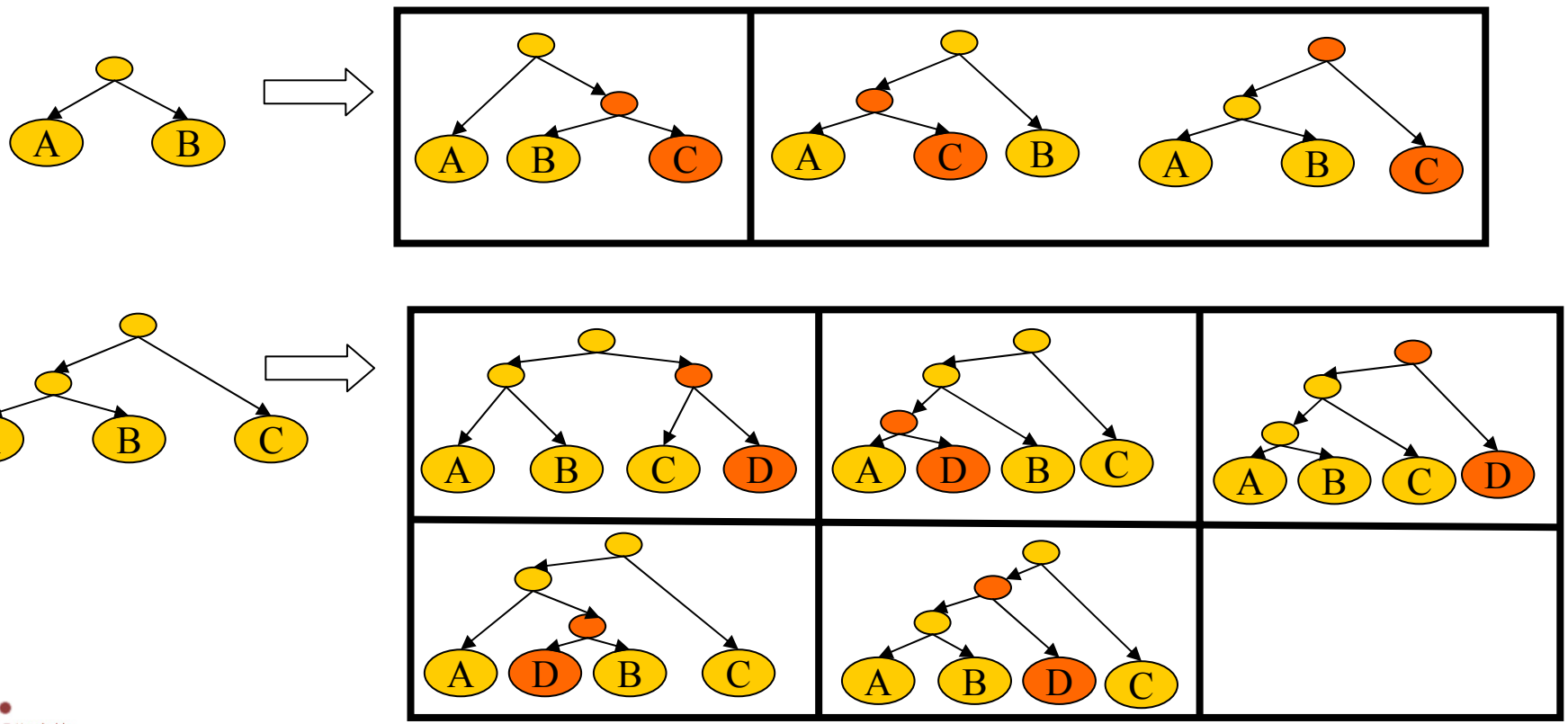
Maximum Parsimony

Ulf Leser

Wissensmanagement in der Bioinformatik

Wie schwierig wird das?

Wie viele binäre, ungeordnete Bäume für n Spezies gibt es?



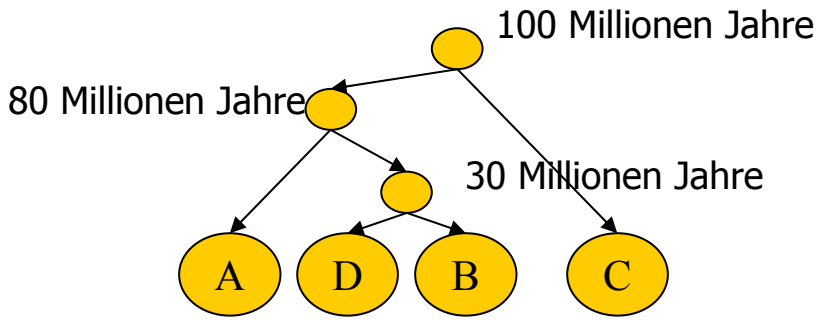
Ergebnis

- Sei t(n) die Zahl binärer Bäume mit n Blättern

$$t(n) = t(2) * t(3) * t(4) * \ldots * t(n-1) =$$
$$= 1 * 3 * 5 * \ldots * (2(n-1)-1) =$$
$$= \frac{(2n-3)!}{2 * 4 * 6 * \ldots * (2n-4)} =$$
$$= \frac{(2n-3)!}{2\left(\frac{2}{2}\right) * 2\left(\frac{4}{2}\right) * 2\left(\frac{6}{2}\right) * \ldots * 2\left(\frac{2n-4}{2}\right)} =$$
$$= \frac{(2n-3)!}{2 * (1) * 2 * (2) * 2 * (3) * \ldots * 2(n-2)} =$$
$$= \frac{(2n-3)!}{2^{n-2} * (n-2)!}$$

1	1
2	1
3	3
4	15
5	105
6	945
7	10.395
8	135.135
9	2.027.025
10	34.459.425
11	654.729.075
12	13.749.310.575
13	316.234.143.225
14	7.905.853.580.625
15	213.458.046.676.875
16	6.190.283.353.629.370
17	191.898.783.962.511.000
18	6.332.659.870.762.850.000
19	221.643.095.476.700.000.000
20	8.200.794.532.637.890.000.000

Ultrametriken



- Stammbaum enthält die „branch points“ der Arten
- 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
  - Wir können den Zeitpunkt der Aufsplitting als Abstandsmaß für zwei Arten (Blätter) benutzen
    - Für Blätter 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)=d(Y,X), und $d(X,Y) \leq d(X,Z)+d(Z,Y)$

Ultrametrische Bäume

- *Definition*
  Sei T ein 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 hat zwei Kinder und ist mit einem Wert aus D beschriftet*
  - *Auf jedem Pfad von der Wurzel zu einem Blatt in T sind die Zahlen 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 echte Stammbaum ist ultrametrisch für die Abstandsmatrix mit den Aufsplittzeitpunkten 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
  - Sprich: 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)
    - ...
  - Begriff: Eine Ultrametrik ist eine Metrik mit der zusätzlichen Bedingung
    $$d(a,c) \leq \max(d(a,b), d(b,c))$$
    - Für Metriken gilt nur $d(a,c) \leq d(a,b)+d(b,c)$

UPGMA Verfahren

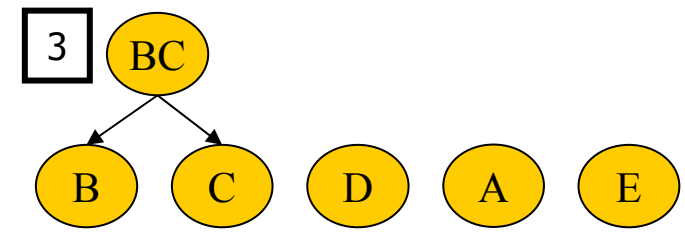
- Gegeben: Distanzmatrix D
- Erzeuge ein „Baumgerüst" mit n Blättern
- Wähle den kleinsten D(i,j) Wert der Matrix und verbinde die Knoten i und j durch einen neuen Knoten (ij) mit Beschriftung D(i,j) und Kanten zu i und zu j
  - Anfangs sind i und j Blätter, später können es auch innere Knoten sein
- Lösche Zeilen und Spalten i und j aus M
- Füge in D eine Zeile und eine Spalte (ij) hinzu mit D(ij,k) = (D(i,k)+D(j,k))/2
- Wiederhole, bis D leer ist

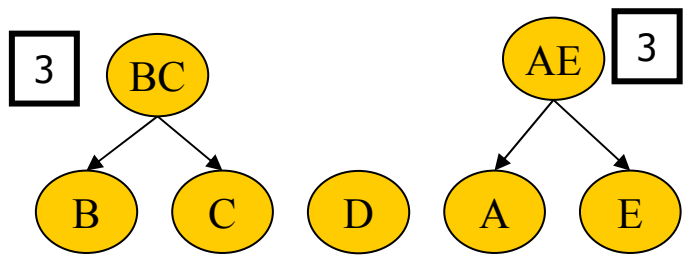
Beispiel

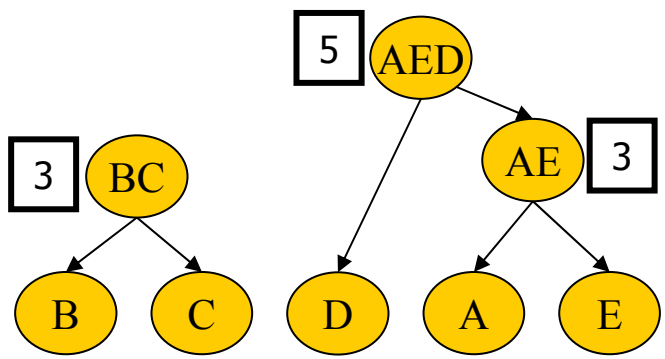
	B	C	D	E
A	8	8	5	3
B		3	8	8
C			8	8
D				5

	BC	D	E
A	8	5	3
BC		8	8
D			5

	BC	D
AE	8	5
BC		8

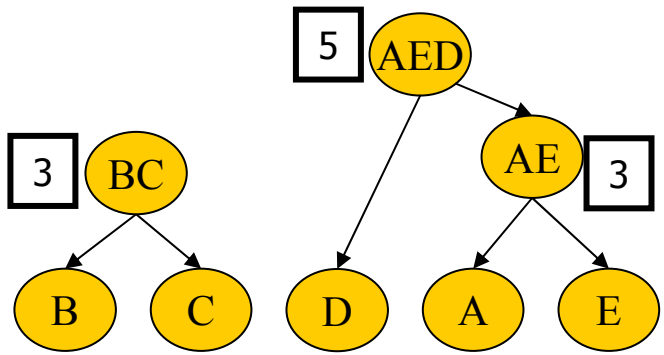




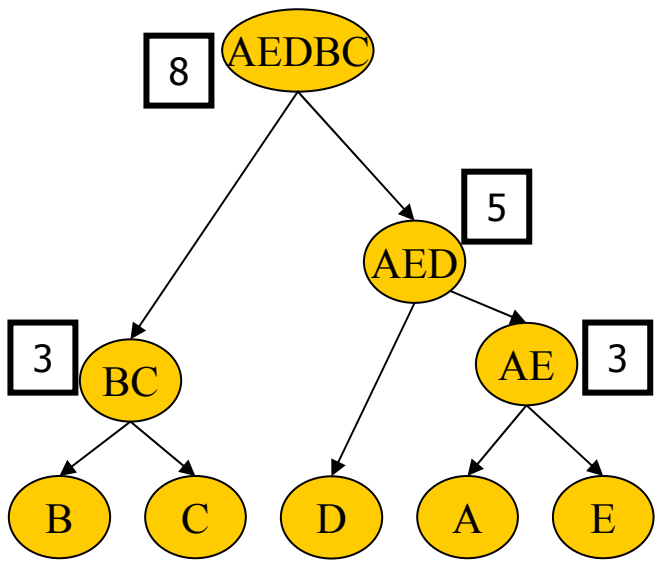


Beispiel

	AE	BC	D
AE		8	5
BC			8



	BC
AED	8



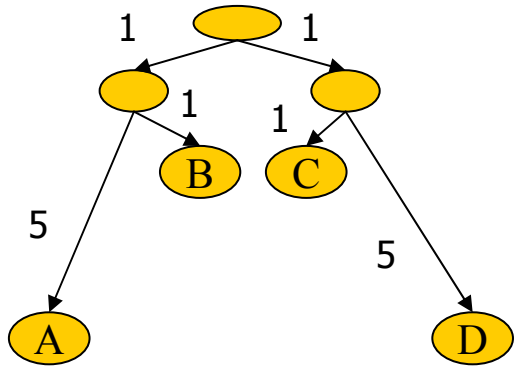
Kontrolle

	B	C	D	E
A	8	8	5	3
B		3	8	8
C			8	8
D				5

Wo UPGMA irrt

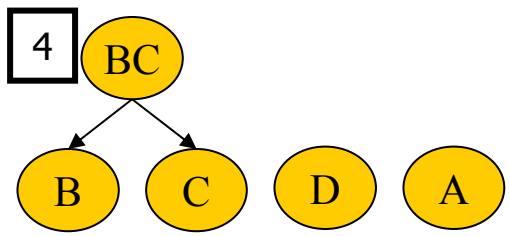
Der echte Baum

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



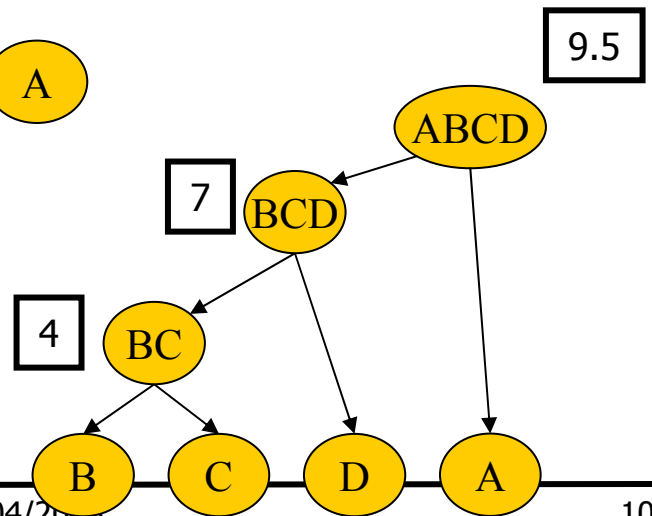
Was UPGMA erzeugt

	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

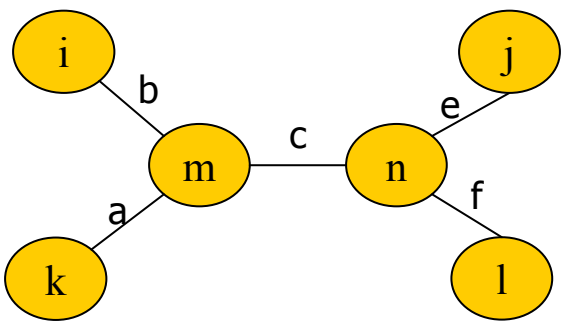
- Gegeben sind keine Zeitpunkte für Speziationsereignisse (innere Knoten), sondern nur wechselseitige Abstände
- Wann können wir daraus einen Baum berechnen?
  - Die Abstände enthalten sich ja gegenseitig
  - Das kann, muss aber nicht zu einem eindeutigen Baum führen

- Definition
  Sei D eine symmetrische Matrix mit n Spalten und Zeilen (und nur positiven Werten; D(i,i)=0). Ein Baum T mit Kantengewichten heißt additiver Baum für D, wenn gilt
  - *T hat n Blätter, beschriftet mit den Zeilen von D*
  - *Für jedes Paar i,j ist D(i,j) gleich der Summe der Kantengewichte auf dem (eindeutigen) Pfad von i nach j*
- Bemerkung
  - Jede ultrametrische Matrix besitzt einen additiven Baum (nämlich den ultrametrischen), aber nicht umgekehrt

Voraussetzungen

- Fragen
  - Existiert zu jeder Matrix ein additiver Baum?
  - Wie findet man einen additiven Baum zu einer gegebenen Matrix?
- *Theorem*
  Eine Matrix D hat einen additiven Baum gdw. für alle Zeilen i, j, k, l ein Baum wie unten angegeben konstruiert werden kann, so dass die 4-Punkt Bedingung gilt:

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



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

- Bemerkung
  - 4-Punkt Bedingung heißt, das Knoten Nachbarn haben, zu denen Sie näher sind als zu Nicht-Nachbarn

Neighbor-Joining

- Matrizen, Eigenschaften, Bäume, 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 nach Nähe zueinander
    - Erzeugt inhärent eine Wurzel (durch wachsenden Abstand)
  - Neighbor Joining
    - Wählt Cluster nach der Nähe zueinander und dem Abstand zu anderen Clustern
      - Sucht (und findet) die nächsten, echten, nahesten Nachbarn
    - Erzeugt einen Baum ohne Wurzel

Inhalt dieser Vorlesung

- Character-basierte Verfahren
- Prinzip des Maximum Parsimony
- Perfect Phylogeny
- Fitch‘s Algorithmus
- Branch & Bound Verfahren
- Steiner Bäume

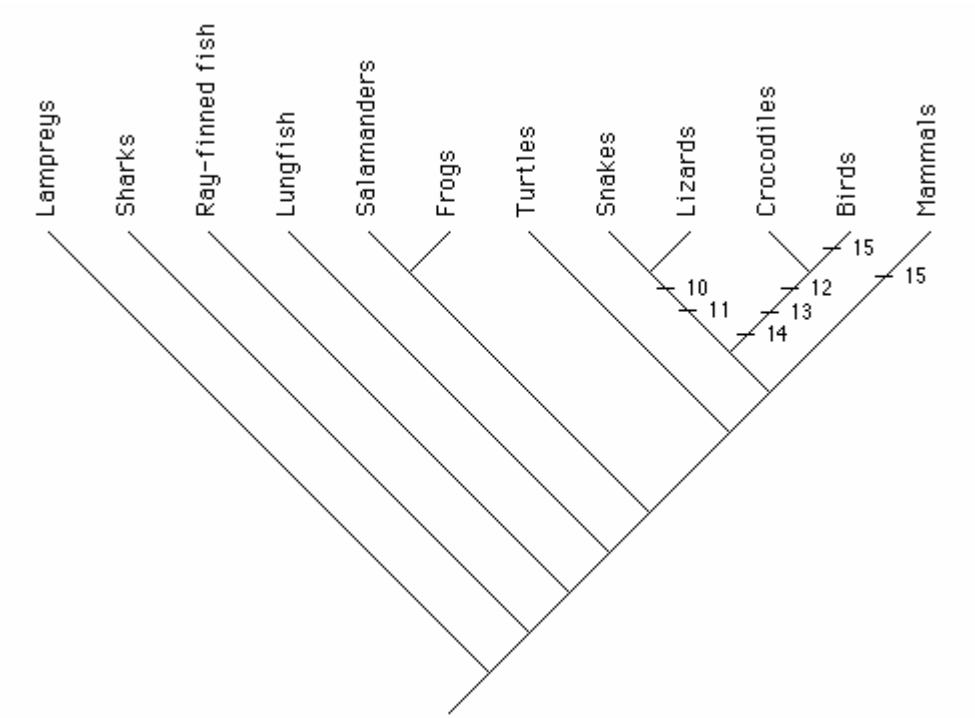
Distanz versus Zeichen

- Distanzbasierte Algorithmen abstrahieren von einzelnen Zeichen und basieren auf dem Abstand von Paaren Taxa (Arten)
  - Ultrametriken, Additive Bäume
- Character-basierte Verfahren betrachten die Entwicklung jedes einzelnen „Characters"
  - Nuklein- oder Aminosäure
  - Morphologische Eigenschaften (Aussehen, Körpergestalt, Organe, ...)
  - Vorhandensein / Abwesenheit bestimmter Gene
  - ...
- Prinzipiell kann man alles benutzen, was in einem Abstammungs-verhältnis steht
  - Sequenzen müssen homolog sein
- Die Auswahl der Character beeinflusst das Ergebnis ganz erheblich
  - Welche Sequenzen? Welche Eigenschaften?
  - Durch die Wahl werden Untergruppen gebildet

Character	Lampreys	Sharks	Teleosts	Lungfish	Frogs	Salamanders	Turtles	Lizards	Snakes	Crocodiles	Birds	Mammals
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



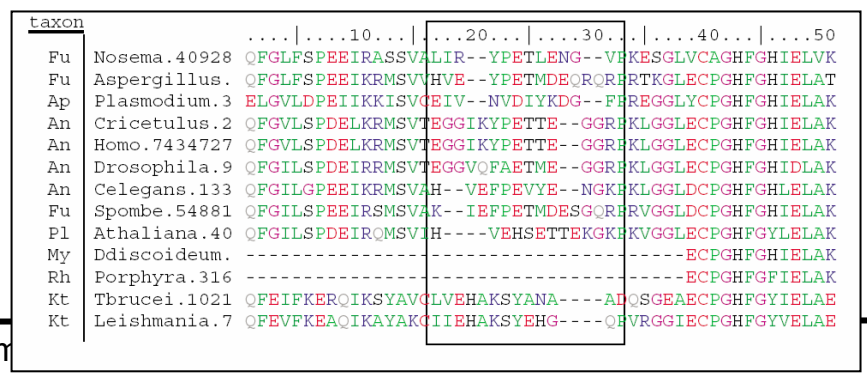
Gesucht: Der Baum mit den wenigsten Änderungen

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
  - Versuche, die Beobachtungen (Arten und Charactere) mit so wenig evolutionären Ereignissen wir möglich zu erklären
- Kriterium, um einen Baum von anderen zu unterscheiden
  - Prinzipiell gibt es unendlich viele Bäume (0-1-0-1-0-1-0-....)
  - Versuch, die Wahl eines bestimmten Baumes „objektiv" zu machen (nachvollziehbar und wiederholbar)

MP und Multiple Sequence Alignment

- Verwendet man Sequenzen, ist jeder Buchstabe (Nukleinsäure, Aminosäure) ein Character
- Welche Basen muss man dabei miteinander vergleichen?
- Vorgehen
  - Multipe Sequence Alignment
  - Spalten mit Insertions und Deletions werden i.d.R. gelöscht
  - Oft: Löschen der Bereiche hoher Heterogenität
- Aber – MSA benötigt oft einen evolutionären Baum
  - Also: Iterieren, auf Konvergenz hoffen



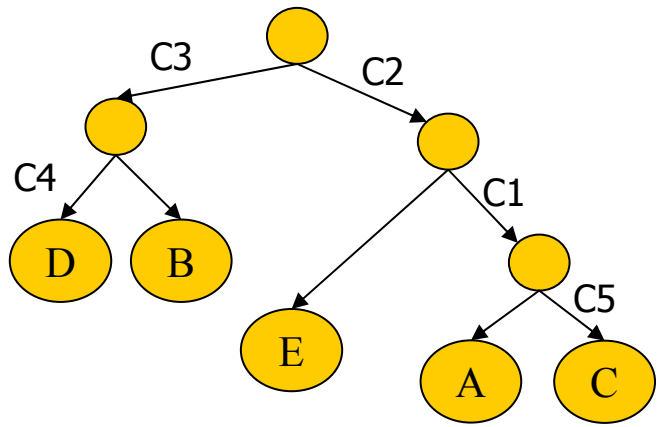
- Perfect phylogeny

Perfect Phylogeny

- Wir beginnen mit einem stark eingeschränkten Modell
  - Jeder Character ist vorhanden oder nicht (binäre Zustände)
  - Die „Urart" (Wurzel) hatte keine der Eigenschaften
  - Jede Eigenschaft ist in der Entwicklung nur genau einmal erzeugt worden
- Wenig realistisch, aber führt zu eleganten Algorithmen
- *Definition*
  - *Sei D eine binäre Matrix aus n Zeilen (den Arten) und m Spalten (den Characters). D(i,j)=1 gdw Art i Eigenschaft j hat*
  - *T ist ein (perfekt-)phylogenetischer Baum für D, wenn gilt*
    - *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 o gilt, dass die Beschriftungen der Kanten auf dem Pfad von der Wurzel zu o genau die Character sind, die o hat*
- Bemerkungen
  - Nicht an jeder Kante von T muss ein Character stehen, aber jeder Character muss an genau einer Kante stehen

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



00000

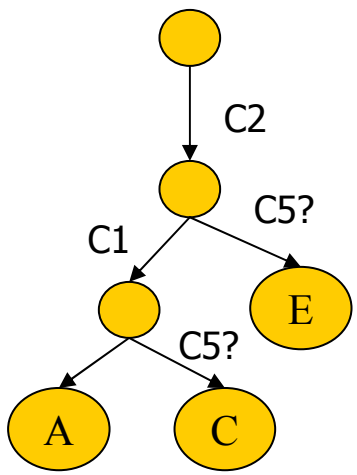
01000

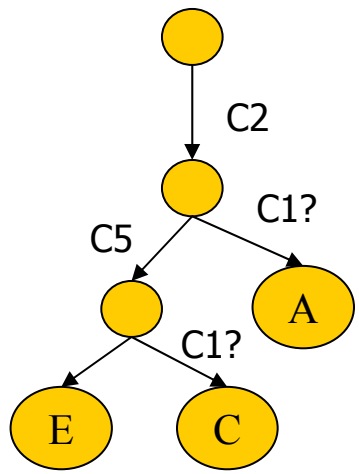
11000

11001

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

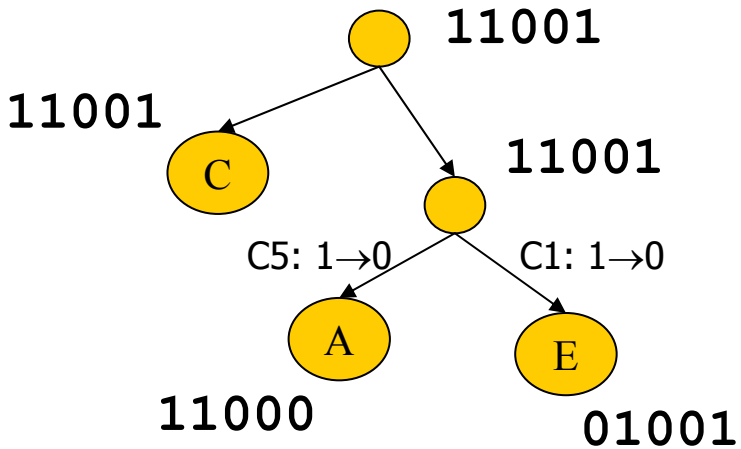




Einschränkungen

- Das sind harte Annahmen
  - Wenn ein Character an mehreren Stellen wechseln darf („convergent evolution"), geht alles gut
  - Wenn Character an der Wurzel nicht abwesend sein müssen und später auch verschwinden können, geht auch alles gut

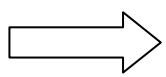
	C1	C2	C3	C4	C5
A	1	1	0	0	0
C	1	1	0	0	1
E	0	1	0	0	1



Existenz perfekt-phylogenetischer Bäume

- *Definition*
  - *Sei D eine binäre Matrix mit n Spalten und m Zeilen. Dann sei D' die Matrix, die aus D wie folgt erzeugt wird*
    - *Interpretiere jede Spalte als binäre Zahl mit der signifikantesten Stelle in Zeile 1*
    - *Sortiere die Spalten in D absteigend nach diesen Zahlen*
  - *Sei D eine binäre nxm Matrix und i eine Spalte. Dann sei O(i) die Menge aller Arten (Zeilen), die die Eigenschaft i besitzen (also in i eine 1 haben)*
- Bemerkung
  - Die Umwandlung D-D‘ ändert nichts an unserem Problem, da alle Character unabhängig und 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*
  Eine transformierte Matrix D' hat einen perfekt-phylogenetischen Baum T gdw. für jedes Paar i, j von Spalten gilt
  - *$O(i) \cap O(j) = \varnothing$*
    - *Artmengen sind disjunkt*
  - *oder $O(i) \subseteq O(j)$ oder $O(j) \subseteq O(i)$*
    - *Eine Artmenge enthält die andere*
- Beweis
  - Richtung „$\Rightarrow$": Sei T ein perfekt-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
    - Beobachtung: Die Menge der Blätter unter e_i bzw. e_j ist exakt O(i) bzw. O(j)
    - Folgende Fälle können in T auftreten
      - $e_i = e_j$: Dann ist O(i)=O(j)
      - e_i liegt auf dem Pfad von der Wurzel zu e_j. Dann gilt $O(j) \subseteq O(i)$
      - e_j liegt auf dem Pfad von der Wurzel zu e_i. Dann gilt $O(i) \subseteq O(j)$
      - e_j und e_i liegen in unterschiedlichen Pfaden. Dann gilt $O(i) \cap O(j) = \varnothing$

Gegenrichtung

- Richtung „$\Leftarrow$“: Sei D‘ eine Matrix mit den genannten Eigenschaften. Wir konstruieren daraus den phylogenetischen Baum T zu D‘ (und zeigen dabei auch Eindeutigkeit des Baumes)
  - Seien p und q zwei Arten (Zeilen) und k die rechteste Spalte mit D‘(q,k)=D‘(p,k)=1. Sei i eine Spalte $i \leq k$
  - Wenn k,i existiert, gilt $O(k) \cap O(i) \neq \emptyset$
    - Denn p und q sind in beiden Mengen
  - Wegen Voraussetzung gilt daher $O(k) \subseteq O(i)$ oder $O(i) \subseteq O(k)$
  - Damit: Wenn D‘(p,i)=1, muss auch D‘(q,i)=1 und umgekehrt
    - Beachte: i ist links von k und D‘ ist auf eine bestimmte Art sortiert

	i	…	k	
B	0	0	1	0
C	0	1	1	0

Ungültig wegen Sortierung von D‘

	i	…	k	
A	1	1	0	0
B	0	0	1	0
C	0	1	1	0

Ungültig, da weder $O(i) \subseteq O(k)$ noch $O(k) \subseteq O(i)$

Gegenrichtung 2

- Wir haben bisher
  - $\forall i \leq k$: D‘(p,i)=D‘(q,i)
  - $\forall j > k$: Entweder D‘(p,j)=D‘(q,j)=0 oder D‘(p,j)≠D‘(q,j)
    - D‘(p,j)=D‘(q,j)=1 verboten wegen Konstruktion von k
- Sei q_S der String der Character von q in der Reihenfolge von D‘ plus ein neues Zeichen „$“
  - Für alle Paare p, q gilt, dass q_S und p_S identisch sind bis zu einer bestimmten Position (k) und danach niemals an derselben Stelle eine 1 haben
  - Wegen „$“ kann kein String einer Zeile Prefix eines anderen sein
- Wir bauen einen Keyword Tree T für alle Strings von Zeilen von D‘
- T ist der 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 1 vor k kann irgendwo sonst im Baum als Kantenbeschriftung auftreten
- qed.

Konstruktion des Baumes

- Komplexität des Existenztests eines phylogenetischen Baums zu D?
  - Wir haben $O(m^2)$ Spaltenvergleiche
  - Und müssen jeweils O(n) Zeilen vergleichen (Enthaltensseinbeziehung)
  - Also $O(nm^2)$
- Wie kompliziert ist es, den phylogenetischen Baum zu D zu konstruieren, 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 Konstruktion des Baums
  - 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 für beliebige z
- 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 Charactern.
  - *Character können Werte aus einer Menge Z mit |Z|=z annehmen.*
  - *Für alle i,j gilt, dass 0<D(i,j)≤z.*

  Ein Baum T mit folgenden Eigenschaften heißt phylogenetischer Baum für D
  - *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: label(k)∈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 T 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
  - Gegeben eine Baumtopologie und eine Taxa/Character Matrix D
  - Bestimmte die Labels der inneren Knoten so, dass der Parsimony Score minimal ist
- Beobachtung
  - Alle Character sind unabhängig voneinander
    - Diese Annahme steckt in dem gesamten MP Ansatz
  - Wenn wir die Topologie festhalten, kann man das small parsimony Problem wie folgt lösen
    - Berechne die besten Label pro Character
    - Setze die Knotenlabel aus den Characterlabeln zusammen
- Also reduziert sich das Problem auf eine Matrix mit einem einzigen Character C mit z Zuständen

Fitch's Algorithmus

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

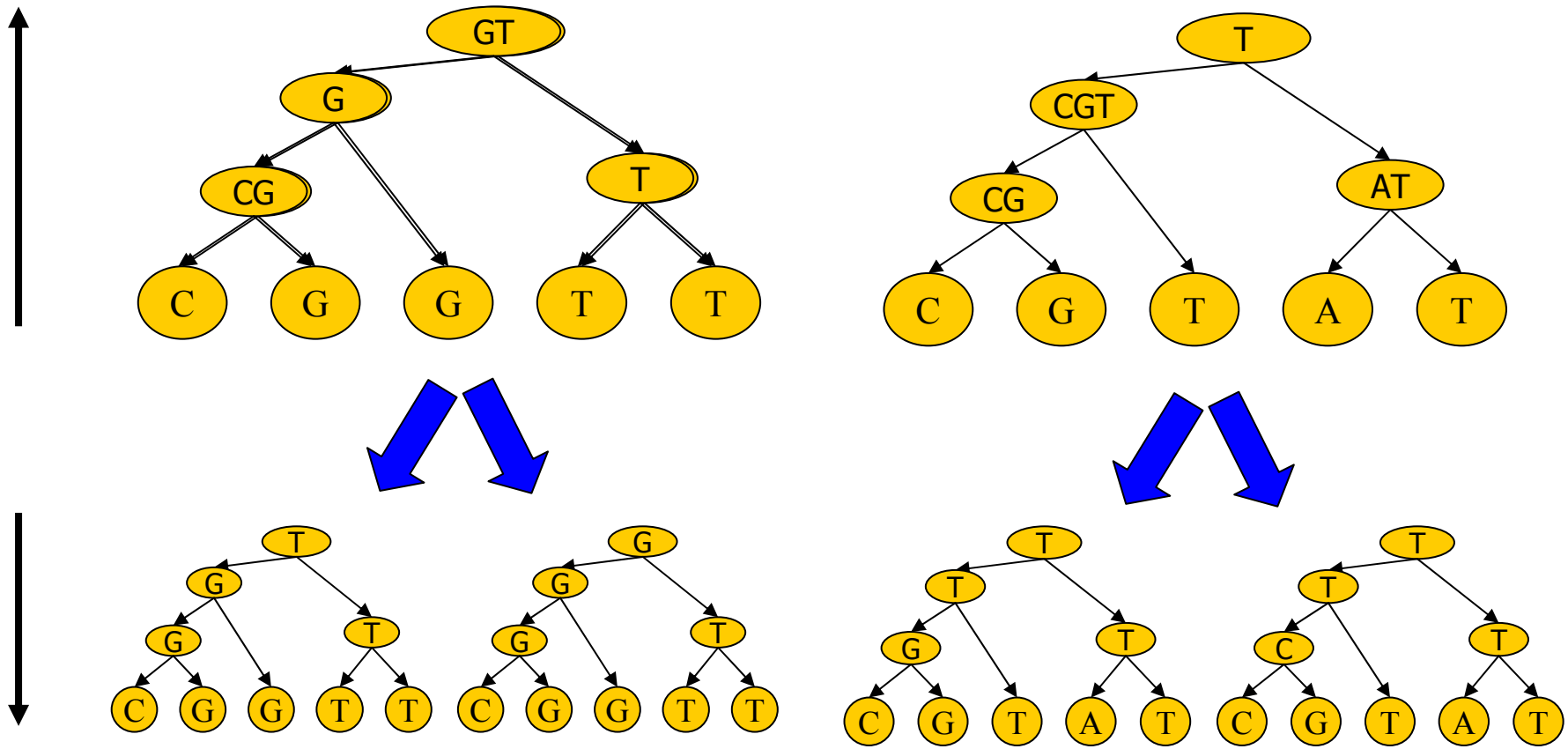
$$P(k) = \begin{cases} P(u) \cap P(v), & falls\ P(u) \cap P(v) \neq \emptyset \\ P(u) \cup P(v) & sonst \end{cases}$$

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

Fitch, Phase 2

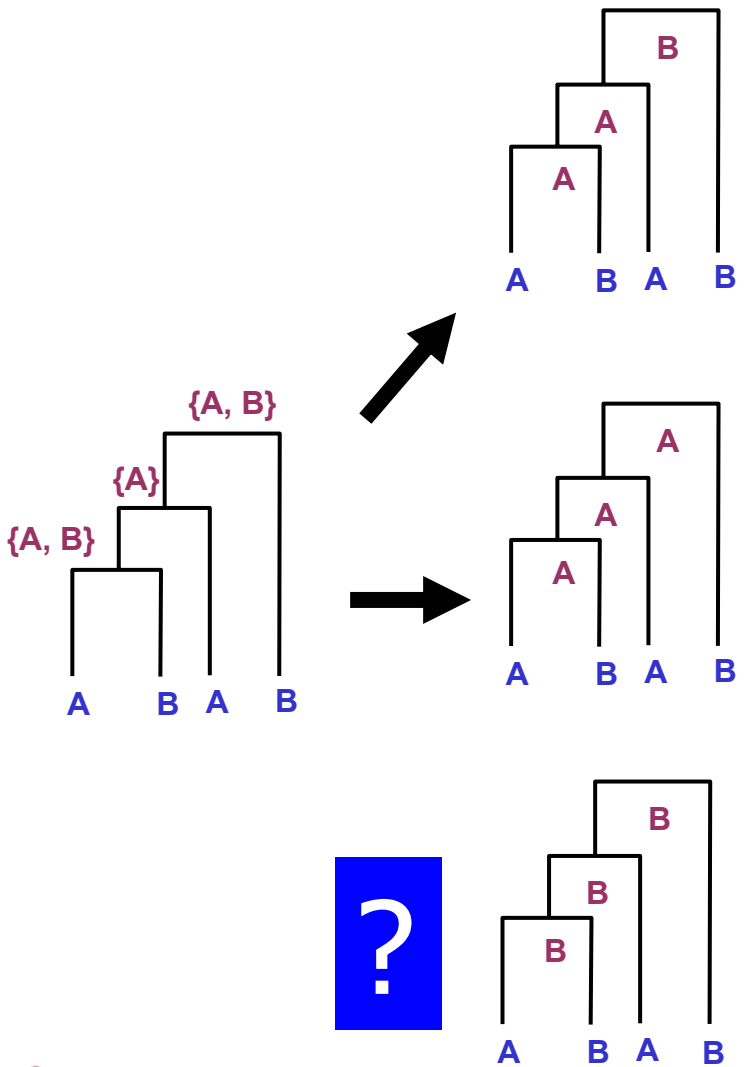
- Alle P(k) sind berechnet
- Phase 2
  - Wähle label(root) beliebig aus P(root)
  - Traversiere alle inneren Knoten k außer root
    - Wenn label(parent(k))$\in$P(k), dann setze label(k)= label(parent(k))
    - Sonst wähle label(k) beliebig aus P(k)
- *Theorem*
  Fitch's Algorithmus berechnet ein Labeling von T mit dem kleinstmöglichen Parsimony Score.
- Beweis
  - Literatur
- Komplexität
  - Phase 1: Für jeden Knoten müssen wir O(z) Vergleiche machen, um P auszurechnen – also O(n*z)
  - Phase 2: z.B. Tiefensuche, linear in der Zahl innerer Knoten, also O(n)
  - Zusammen: O(n*z)

Beispiel



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

Aber Vorsicht



- Fitch's Algorithmus findet nicht alle optimalen Bäume
  - Alg. ist im Prinzip 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)
  - Das entspricht nicht der Realität
  - Siehe PAM, BLOSSUM, etc.
- Weighted Parsimony
  - Übergänge werden individuell gewichtet
  - Verwendung einer Substitutionsmatrix S
  - S(i,j) = Kosten um Character von Zustand i in Zustand j zu ändern
- Problemformulierung
  - Gegeben eine Baumtopologie T und eine Matrix D
  - Finde eine Beschriftung der inneren Knoten von T so, dass der folgende Ausdruck minimiert wird

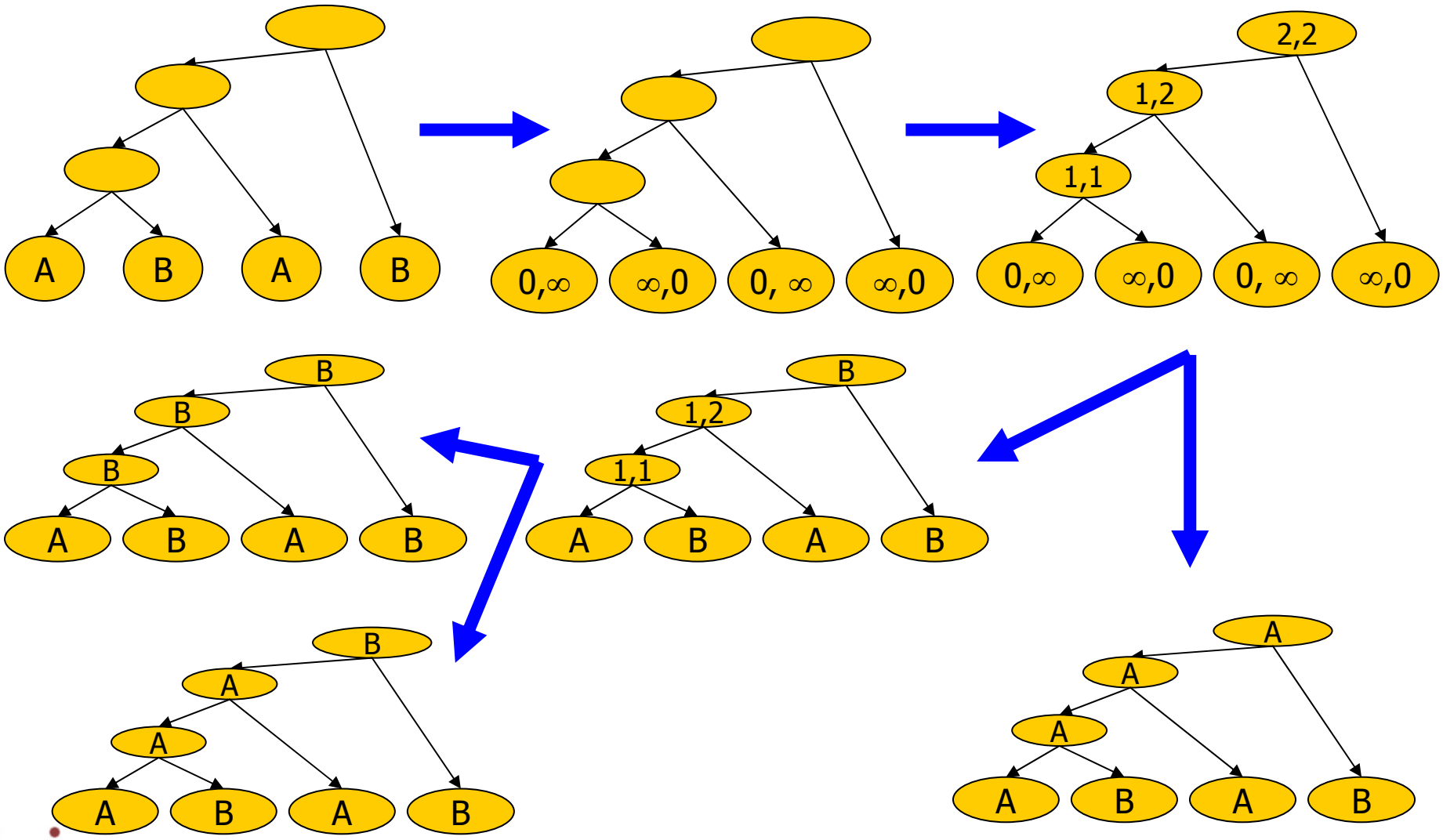
$$S^w(T) = \sum_{(u,v)\in E(T)} S(label(v), label(u))$$

Sankoff's Algorithmus

- Zwei Phasen wie bei Fitch
  - 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 $$S_z(k) = \begin{cases} 0, wenn\ label(k) = z \\ \infty\ sonst \end{cases}$$
  - Laufe durch den Baum und berechne für jeden Knoten k mit Kindern u und v (i läuft über alle Zustände):
$$S_z(k) = \min_i(S(z,i) + S_i(u)) + \min_i(S(z,i) + S_i(v))$$
- Phase 2
  - Wurzel: $$label(root) = \min_i(S_i(root))$$
  - Knoten k mit Vater u: $$label(k) = \min_i(S(label(u),i) + S_i(k))$$

Beispiel – Was bei Fitch schief ging

	A	B
A	0	1
B	1	0

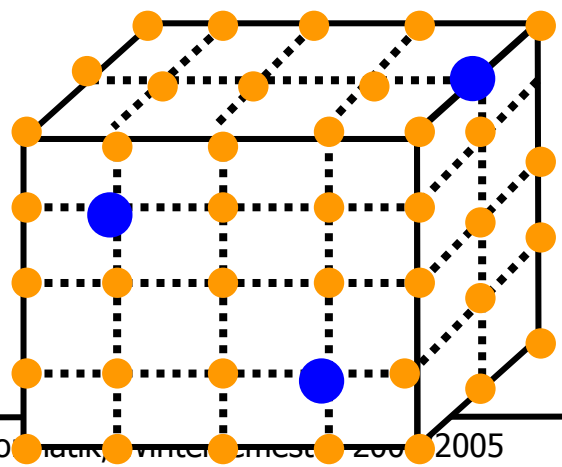


Large parsimony

- Small parsimony kann man also effizient lösen
- Das gilt aber nur für eine feste Baumtopologie
- Large Parsimony
  - Gegeben eine Matrix D. Finde die Baumtopologie und Beschriftung der inneren Knoten so, dass der parsimony score minimal ist über alle Topologien und Beschriftungen.
- Unglücklicherweise gilt
  - Das „large parsimony" Problem ist NP-vollständig
- Im Prinzip müssen wir also alle möglichen Topologien ausprobieren
- Warum ist das so?

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
  - Jedes Taxa ist ein Punkt in diesem Raum
  - Der Abstand zweier Taxa ist „ungefähr" der Manhattan Abstand der Punkte
    - „Ungefähr", weil es egal ist, wie weit man auf einer Achse geht
  - Das MP Problem ist äquivalent zu: Finde den Steiner Baum für alle Taxa-Punkte im Graphen G; G besteht aus allen Gitterpunkten des Raumes und Kanten entlang aller Achsen



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 dabei immer die optimalen Kosten 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 2k-1 Möglichkeiten
  - Halte eine fest und steige weiter ab (5. Art, 6.Art ...)
- Am Blatt des Suchbaums haben wir eine komplette Topologie für D mit optimalem Score S
  - S ist garantiert nicht kleiner als irgendeiner der bisher berechneten Scores
- Traversiere den Rest des Baums
  - Immer, wenn ein (Teil-)Baum einen Score größer S hat, vergiss diesen Art des Suchraums (Pruning)
  - Passe S jeweils an das aktuelle Optimum an

Eigenschaften

- Vergleichbar dem A* Algorithmus
- Die Worst-Case Complexität ist immer noch exponentiell in der Anzahl Taxa
- Aber „normale" Laufzeiten sind deutlich besser
- Man kann viele Heuristiken hinzufügen
  - Mit welchen Arten soll man beginnen? Möglichst ähnlichen?
  - Welche Bäume soll man zuerst aufzählen?
  - ...

Andere Möglichkeiten

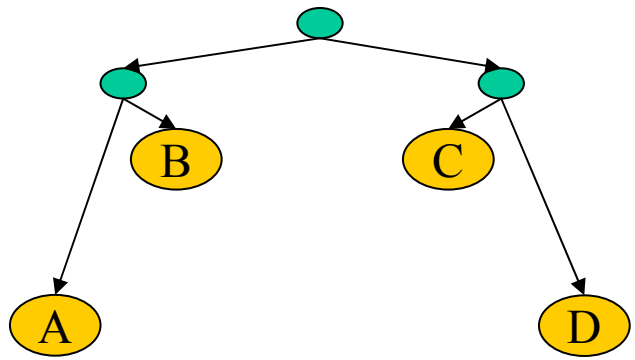
- Iterative Verbesserung
  - Beginne mit irgendeiner Topologie und berechne optimalen Score
  - Verändere diese „lokal", nach Möglichkeit zum Guten
  - So lange, bis es nicht mehr besser wird
  - Wiederhole das mit vielen zufälligen Startbäumen
  - Der beste gefundene Baum ist das Ergebnis
  - Keine Garantie für Finden des globalen Optimum
- Greedy
  - Zähle alle Bäume mit wachsender Größe auf
  - Berechne für eine Topologie jeweils den besten Score
  - Wähle jeweils den besten Baum, und erweitere nur diesen um die nächste Art

Weitere Beobachtung

- Löschen von „uninformative characters“
  - Wir können aus der Matrix alle Spalten löschen, für die gilt
    - Alle Werte sind gleich (da ändert sich nie was)
    - Jeder Wert kommt nur einmal vor (das kostet z Änderungen, egal wie der Baum aussieht)
    - Es gibt keine zwei Zustände, die mindestens zweimal vorkommen (z.B: XXXYZ) – das kostet 2, egal wie man es dreht
  - Dadurch wird das Small Parsimony Problem billiger
  - Aber die Komplexität 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 richten 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 die Meinung, dass alle Methoden recht gut funktionieren
  - Gilt nur bei einfachen Evolutionsmodellen
  - Güte hängt von den Eigenschaften der Daten ab
- Distanzbasierte Methoden
  - Am ungenauesten
  - Dafür sehr schnell
  - Brauchen numerische Abstandsmasse
- Maximum Parsimony
  - Berücksichtigt mehr Informationen
  - Bessere Methode (aber nicht optimal)
  - Optimale Lösung für größere Probleminstanzen nicht mehr lösbar
- Typisch: alle Methoden parallel angewanden und die Ergebnisse vergleichen
  - Gruppen, die überall gleich vorkommen, gelten als sehr robust
  - „Consensus tree"