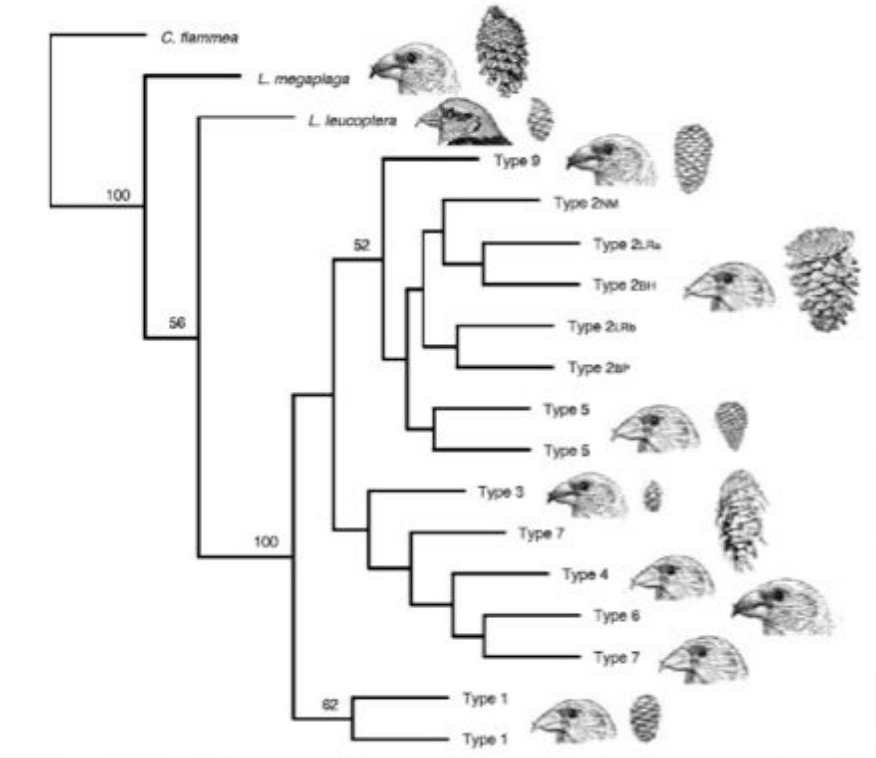
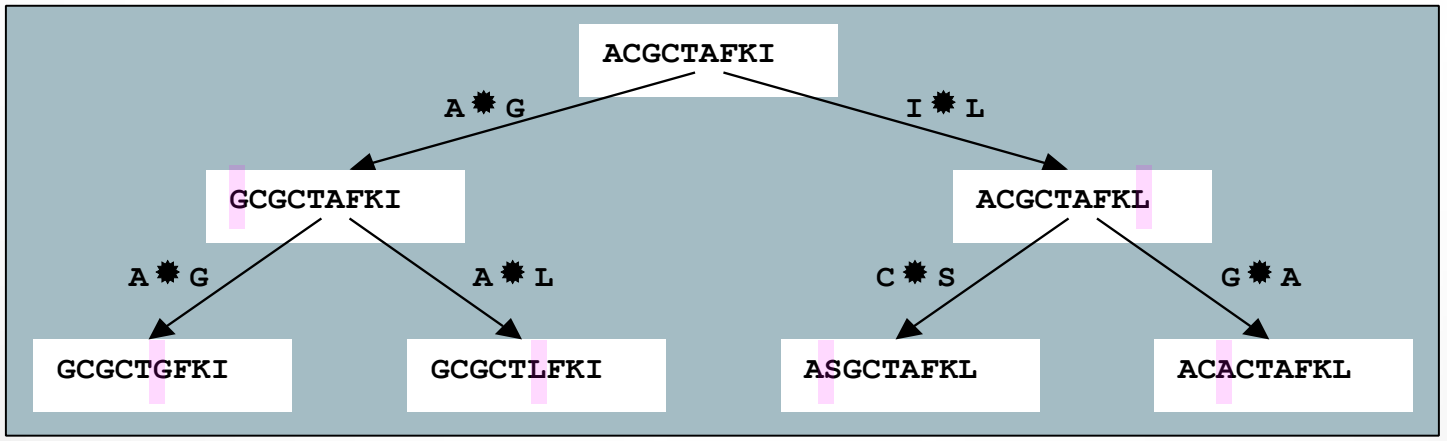
Simboliais grįsta filogenezė



- Simboliais grįsta filogenezė
  - **Didžiausio taupumo (Maximum Parsimony)**
  - Didžiausios tikimybės (Maximum likelihood)

Didžiausios tikimybės metodai (maximum likehood)

- Sukurti visus įmanomus medžius
- Surasti medžių tikimybę
  - Galima naudoti pakeitimų tikimybes (pvz. Jukes-Cantor)
- Pasirinkti medį su didžiausia tikimybe



- **Labai lėtas, didelė paieškos apimtis**
- **Reikia ieškoti bendrų protėvių**

Didžiausio taupumo metodas (maximum parsimony)

Ieškome medžio, kurį paaiškintų mažiausias pasikeitimų skaičius.

Palyginimo matrica vs. Atstumų matrica

- Tarkim turime geno iš *m* nukleotidų seką ir *n* rūšių dygybinio palyginimo gauname *n* x *m* palyginimo matricą (daugybinį palyginį).

- Gaime daugybinį palyginį transformuoti į porinių atstumų matricą.

- Tačiau negalime iš atstumų matricos gauti sekų palyginio ir informacija yra prarasta.

Simbolių analize paremta filogenezė

- Geresnė metodika: Simbolių analize pagrįsti algoritmai naudoja nadoja *n* x *m* palyginio matricą arba daugybinį palyginį.
  - (*n* = rūšių skaičius, *m* = simbolių skaičius)
  - Naudojamas sekų palyginimas vietoje atstumų matricos.
  -
- Tikslas - nustatyti kokia simbolių eilutė geriausia atitiktų vidinius medžio mazgus visoms n stebimos sekos apjungtoms į medį.

Simbolių analize paremta filogenezė

- Simbolis gali būti nukleotidai A, G, C, T, kurie yra simbolio būsenos. Arba aminorūgštys arba kiti simboliai žymintys akių, kojų skaičių arba snapo ar žvynų forma.
- Medžio kraštinių ilgiai proporcingi simbolių pakeitimams reikalingiems paaiškinti sudaromą filogenetinį medį ~ atstumams tarp sekų (mazgas - seka). Atstumas tarp mazgų yra proporcingas simbolių pakaeitimų skaičiui tarp atitinkamų sekų. Visam medžiui galima suskaičiuoti bendrą tokių atstumų sumą - tai vadinamas **taupmo įvertis.**

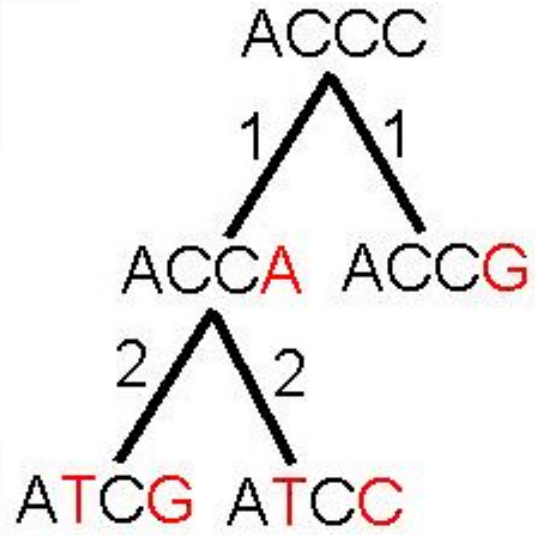
Maža taupumo problema (Parsimony - taupumas)

Taupumo metodas kuriant filogenetinius medžius

- **Taupumas**: taiko Occam'o skustuvo principą: ir iš visų galimų filogenetinių medžių pasirenkam tą, kuris būtų paprasčiausias.
  - Taupumo metodas laikosi prielaidos, kad stebimi skirtumai tarp sekų atsirado dėl mažiausio įmanomo mutacijų skaičiaus
  - Ieškomas medis, kurio **taupumo įvertis** būtų pats mažiausias iš visų galimų medžių. T.y. jis atitiktų mažiausią galimą mutacijų skaičių.

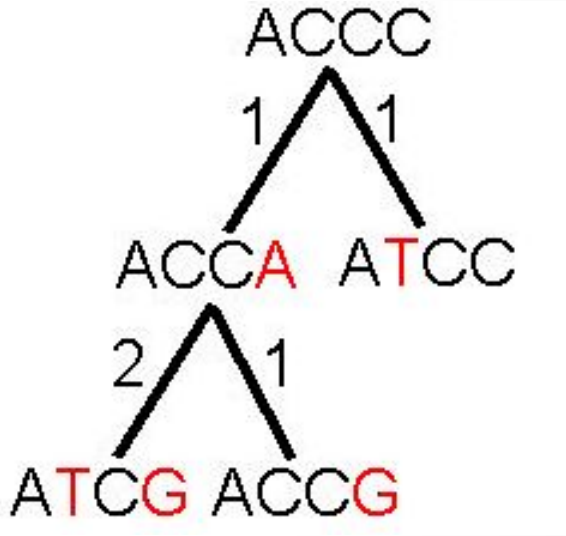
Taupumo metodas kuriant filogenetinius medžius

Mažiau Taupus



Įvertis= 6

Labiau Taupus



Įvertis= 5

Maža taupumo problema

- Įvestis: Medis *T* , kurio kiekvienas lapas (išorinis mazgas atitinkantis dabar egzistuojantčią realią seką) *m*-simbolių eilutę..

- Rezultatas: Vidinių mazgų sekų radimas , kurie minimizuotų taupumo įvertį.

- Laikome, kad kiekvienas lapas žymimas tik vienu simboliu. Nes laikomes prielaidos, kad simbolis sekoje kinta nepriklausomai nuo kitų simbolių.

Svorinė maža taupumo problema

- Išplėsta mažos taupumo problemos versija.
- Įvestis: *k* x *k* įverčių matrica, nurodanti veino simbolio virtimo kitu "kainą".
- Mažos taupumo problemos atveju įverčių matrica atitinka paprasčiausią DNA atstumą (Hamingo):

$$d_H(v,w) = \begin{cases} 0 & \text{if } v = w \\ 1 & \text{otherwise} \end{cases}$$

Įverčių matricos: Pavyzdys

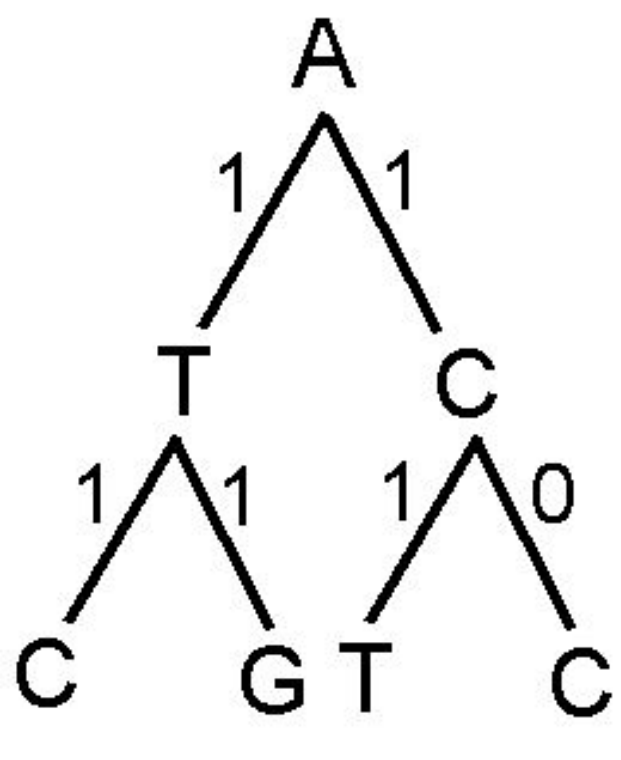
Maža taupumo problema

	A	T	G	C
A	0	1	1	1
T	1	0	1	1
G	1	1	0	1
C	1	1	1	0

Svorinė taupumo problema

	A	T	G	C
A	0	3	4	9
T	3	0	2	4
G	4	2	0	4
C	9	4	4	0

Nesvorinė vs. Svorinė

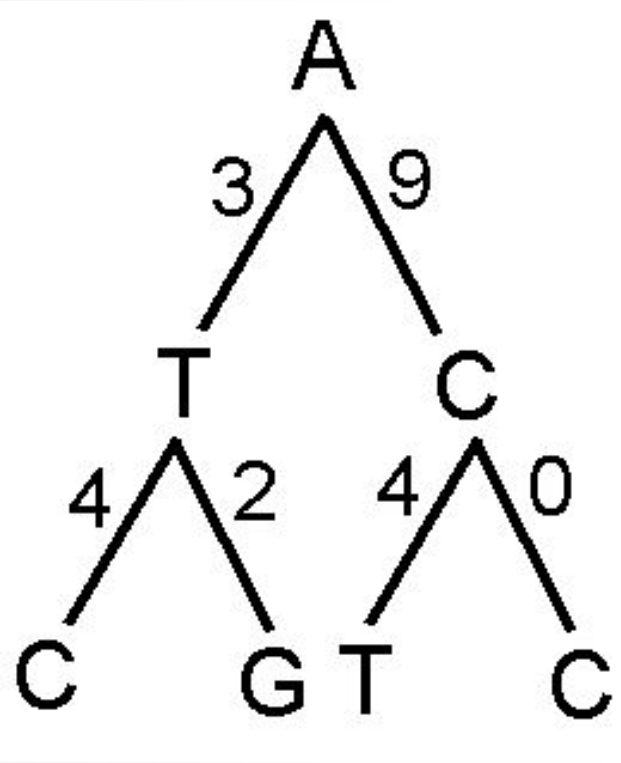


Mažos taupumo problemos įverčių matrica

	A	T	G	C
A	0	1	1	1
T	1	0	1	1
G	1	1	0	1
C	1	1	1	0

Taupumo įvertis 5

Nesvorinė vs. Svorinė



Svorinės taupumo problemos įverčių matrica

	A	T	G	C
A	0	3	4	9
T	3	0	2	4
G	4	2	0	4
C	9	4	4	0

Svorinio taupumo įvertis: 22

Svorinė maža taupumo problema

- Įvestis: medis T, kurio kiekvienas lapas pažymėtas elementais iš k-raidžių alfabeto su įverčių matrica k x k ($\delta_{i,j}$)

- Rezultatas: Vidinių mazgų sekų radimas , kurie minimizuotų taupumo įvertį.

Fitch ir Sankoff algoritmai

Sankoff's Algritmas. Dinaminis programavimas

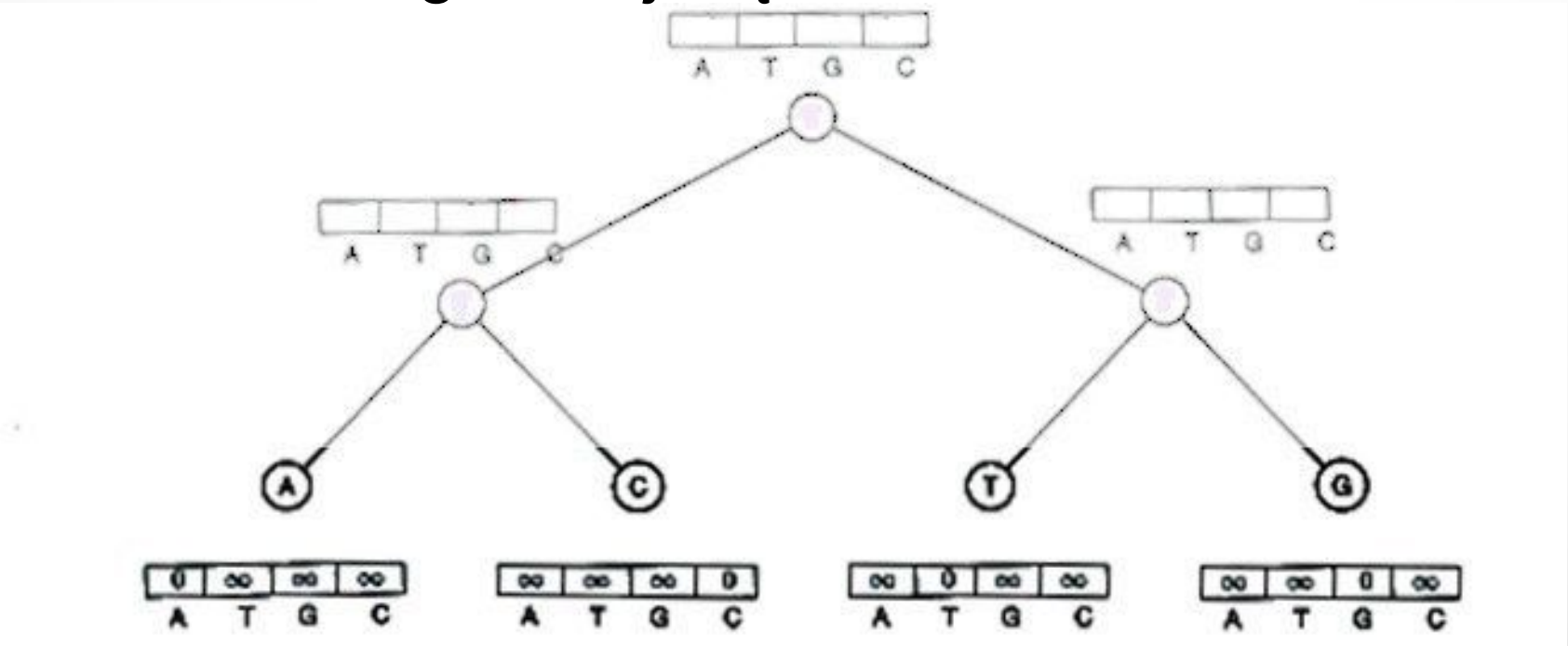
- Apskaičiuok ir sek įverčius visių galimų mazgo verčių (simbolio ar eilutės) kiekvienam mazgui.
  - $s_t(v)$ = minimalus dalinio medžio (dalinio - analizuojant tik mazgus kurie atsišakoja nuo v mazgo), kurio šaknis priskirta *v* mazgui, taupumo įvertis jei v mazgas turi simbolį *t* .
- Kiekvieno mazgo įvertis priklauso nuo jo vaikų įverčių.

$$s_t(\text{parent}) = \min_i \{ s_i(\text{left child}) + \delta_{i,t} \} + \min_j \{ s_j(\text{right child}) + \delta_{j,t} \}$$

- Taigi viso medžio taupumo įvertis bus jo šaknies ar šakninio mazgo įvertis.

Sankoff Algritmas

- Pradedam ties lapais:
  - Jei lapas turi analizuojamą simbolį, tuomet jo įvertis tegul būna 0.
  - Priešingu atveju , įvertus ∞.



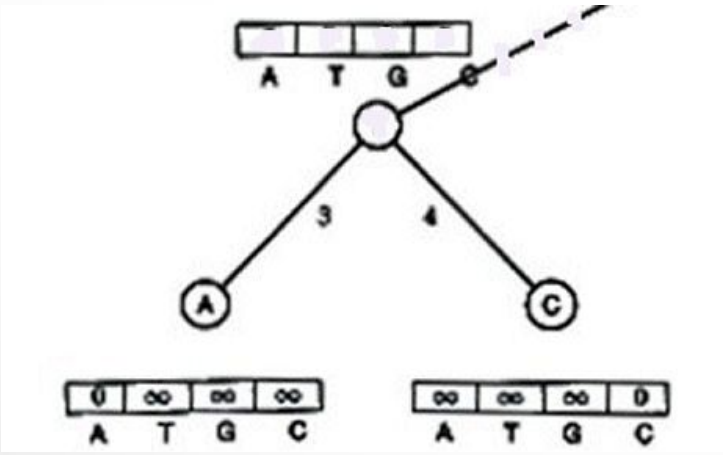
Sankoff Algritmas

- **Pritaikome dinaminį programavimą kartu su įverčių matrica pakilti nuo lapų į aukštesnį lygmenį.**

δ	A	T	G	C
A	0	3	4	9
T	3	0	2	4
G	4	2	0	4
C	9	4	4	0

$$s_t(v) = \min_i \{ s_i(u) + \delta_{i,t} \} + \min_j \{ s_j(w) + \delta_{j,t} \}$$

$$s_A(v) = 0 + \min_j \{ s_j(w) + \delta_{j,A} \}$$

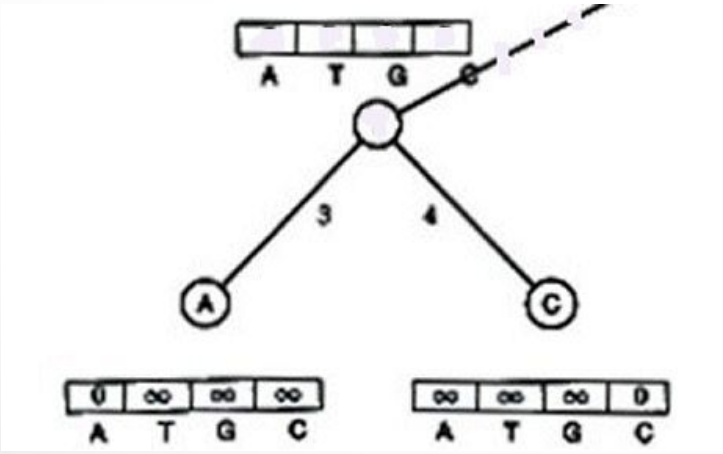


	$s_i(u)$	$\delta_{i,A}$	suma
A	0	0	0
T	∞	3	∞
G	∞	4	∞
C	∞	9	∞

Sankoff Algritmas

- **Pritaikome dinaminį programavimą kartu su įverčių matrica pakilti nuo lapų į aukštesnį lygmenį.**

δ	A	T	G	C
A	0	3	4	9
T	3	0	2	4
G	4	2	0	4
C	9	4	4	0



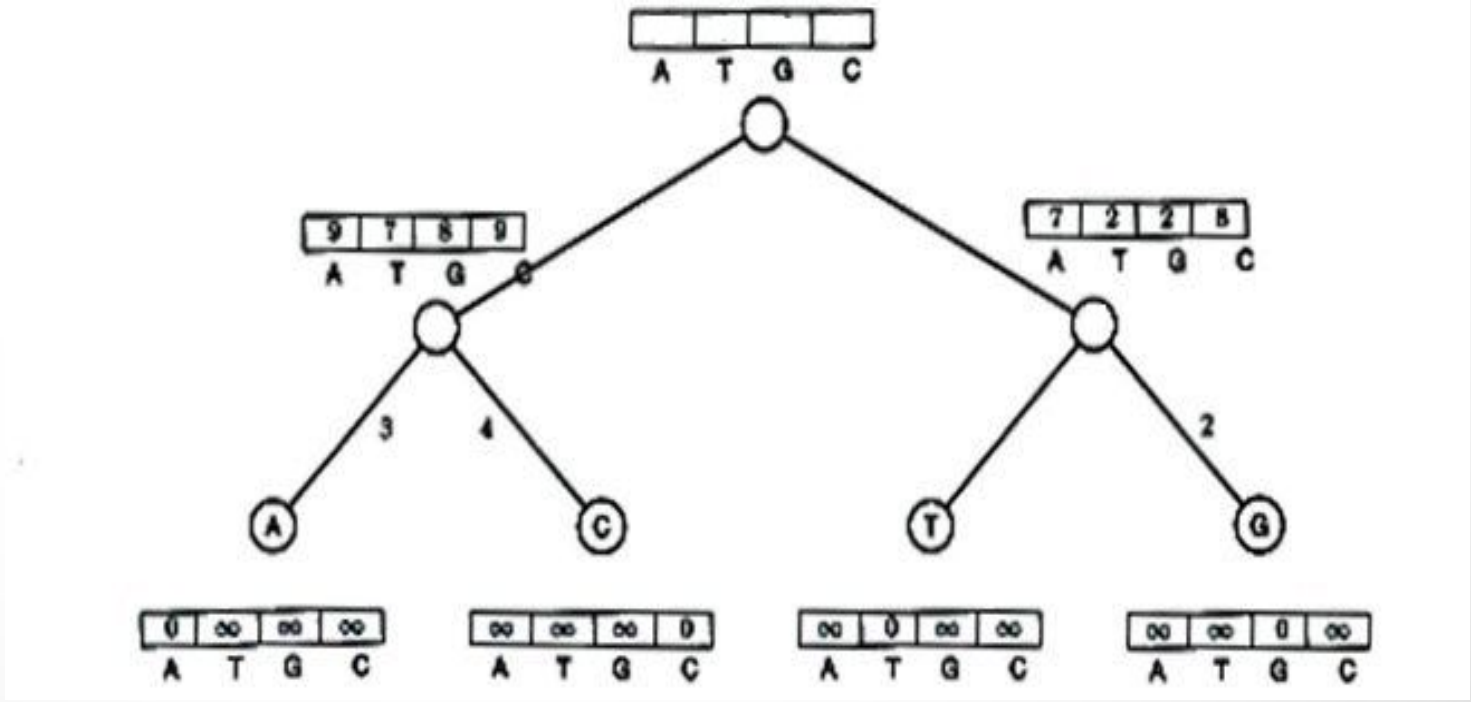
$$s_t(v) = \min_i \{s_i(u) + \delta_{i,t}\} + \min_j \{s_j(w) + \delta_{j,t}\}$$

$$s_A(v) = 0 + 9 = 9$$

	$s_i(u)$	$\delta_{i,A}$	suma
A	0	0	0
T	∞	3	∞
G	∞	4	∞
C	∞	9	∞

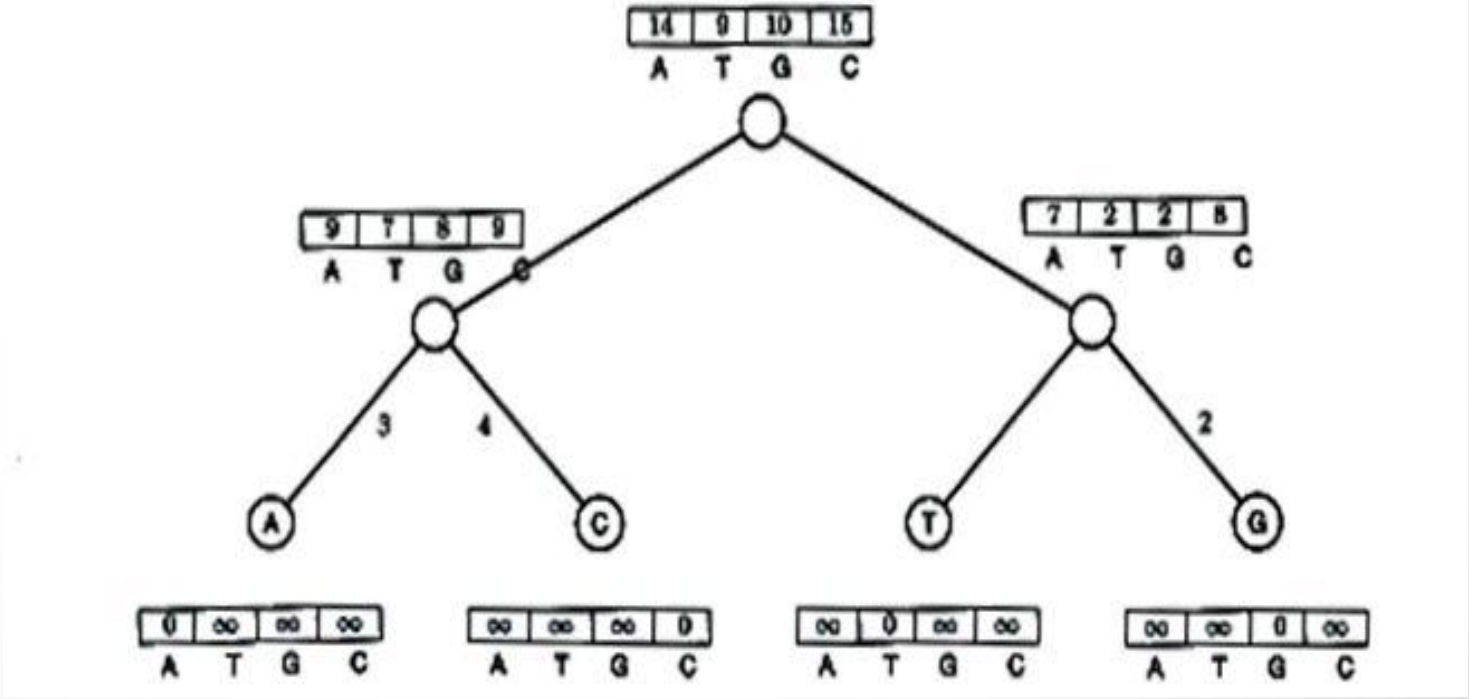
Sankoff Algritmas

- Pakartojam dešiniam daliniam medžiui:



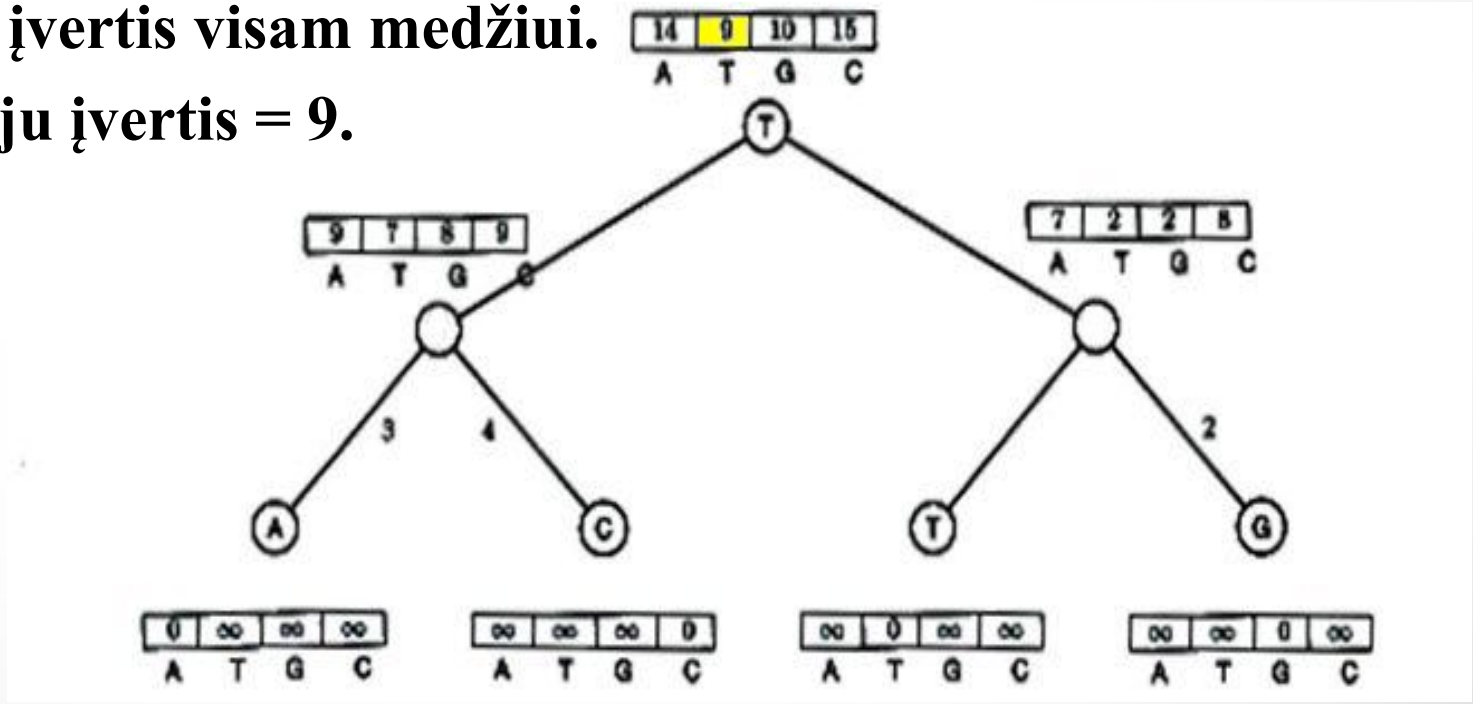
Sankoff Algritmas

- **Pakartojam dešiniam daliniam medžiui:**
- **Pakartojam šakniai:**



Sankoff Algritmas

- **Pakartojam dešiniam daliniam medžiui:**
- **Pakartojam šakniai:**
- **Mažiausias šaknies įvertis yra mažiausias svorinio taupumo įvertis visam medžiui.**
- **Šiuo atveju įvertis = 9.**



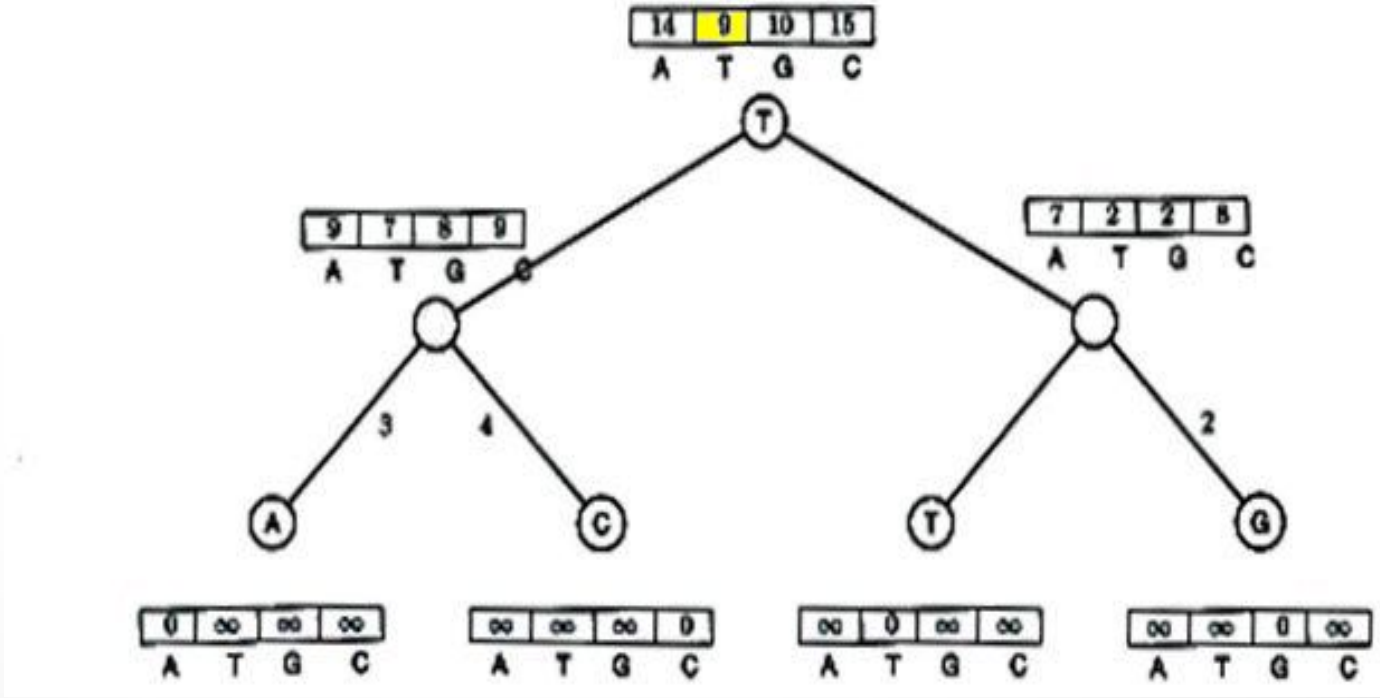
Sankoff Algoritmas: Keliavimas medžiu žemyn

- Įverčiai ties šakniniu mazgu visoms galimoms šio mazgo vertėms yra apskaičiuoti kylant medžiu į viršų

- Po šio etapo Sankoff algoritmas keliauja mazgais žemyns ir priskiria optimalų simbolį kiekvienam mazgui.

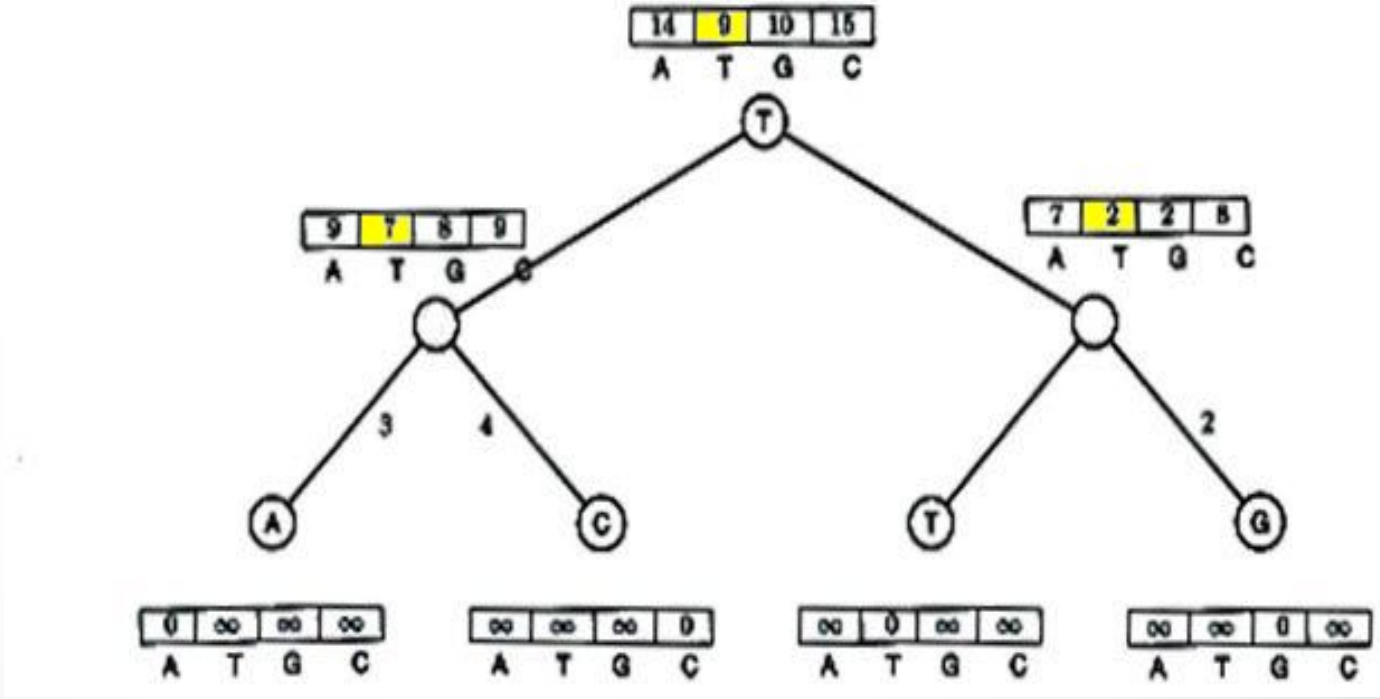
Sankoff Algritmas

- 9 yra iš 7 + 2.



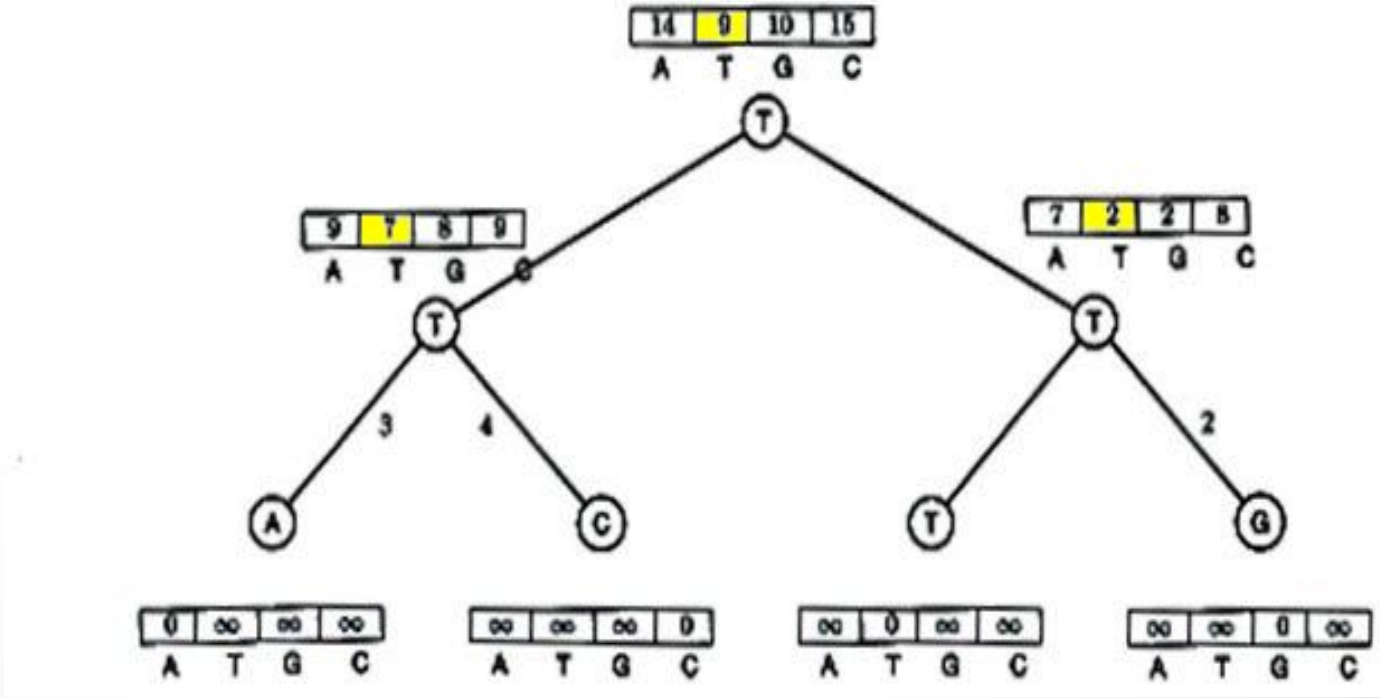
Sankoff Algritmas

- 9 yra iš 7 + 2.
  - Taigi kairysis vaikas yra T, ir dešinysis yra T.



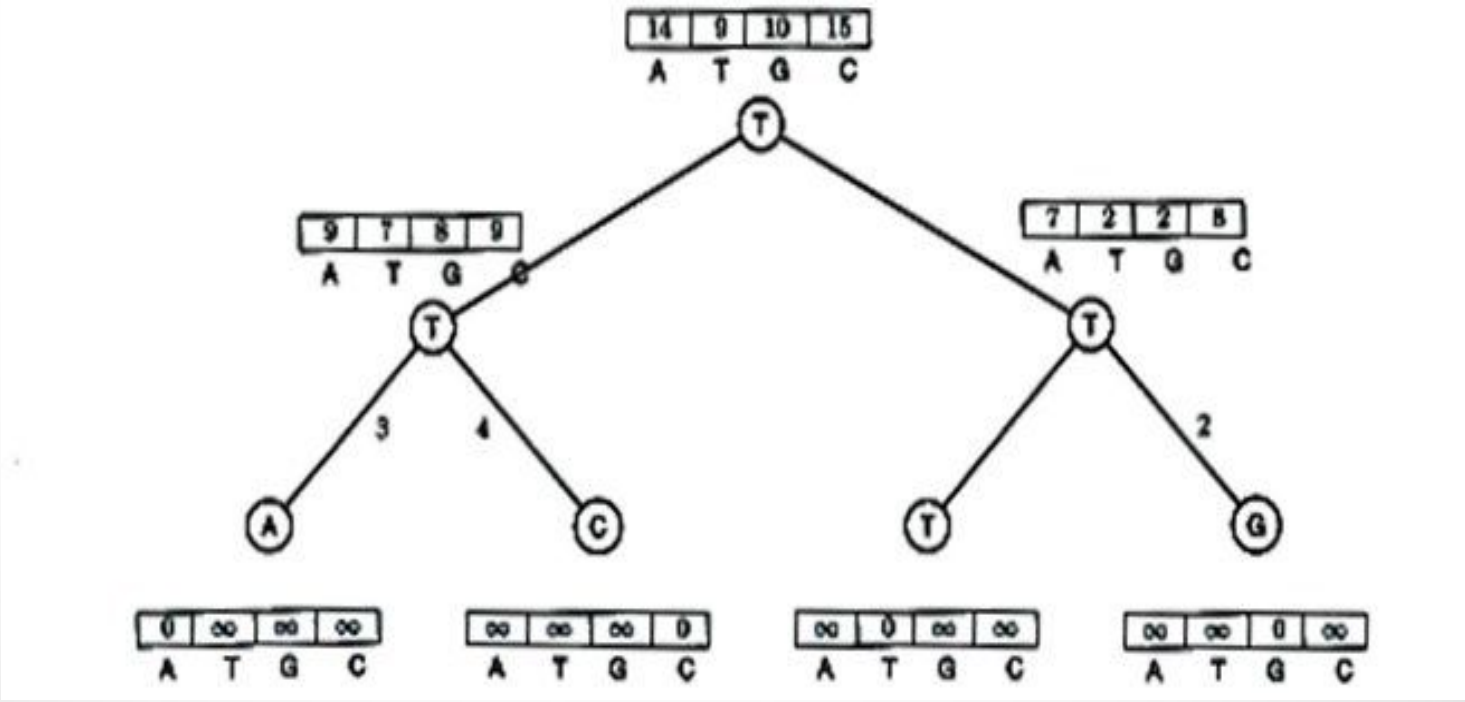
Sankoff Algritmas

- 9 yra iš 7 + 2.
  - Taigi kairysis vaikas yra T, ir dešinysis yra T.



Sankoff Algritmas

- 9 yra iš 7 + 2.
  - Taigi kairysis vaikas yra T, ir dešinysis yra T.
- Priskirti sibloliai visam medžiui...



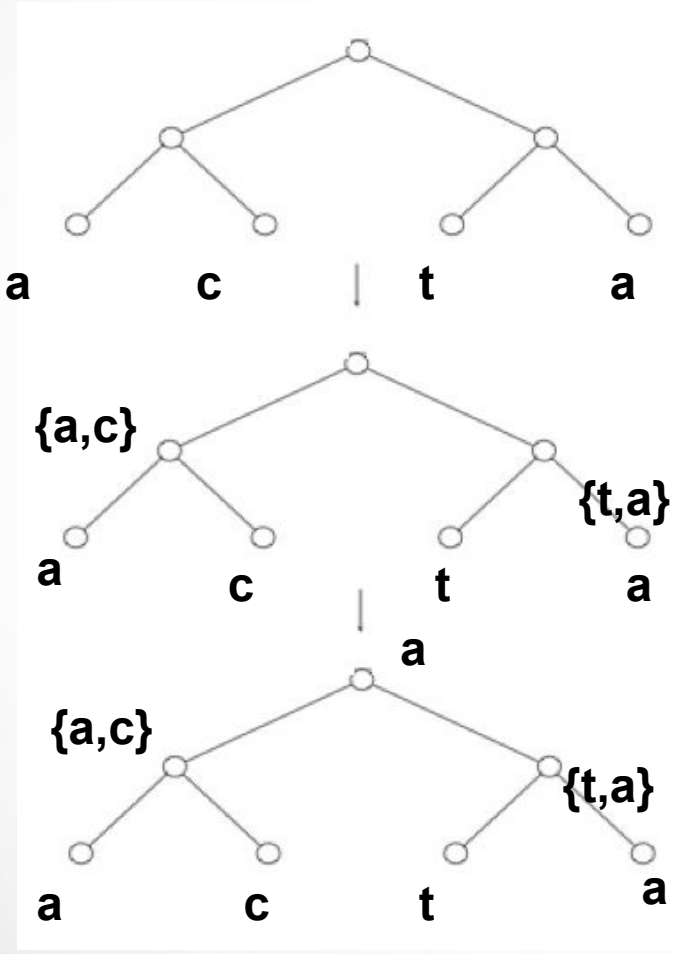
Fitch Algritmas

- Sprendžia mažą taupumo problemą.

- Kiekvienam mazguo priskiria simbolių rinkinį.
  - Kiekvinam lapui priskiriamas jame stebimas simbolis (lapai dabar egzistuoja tai simbolis - duotybė)

- Kylant medžiu aukštyn nuo lapų, kiekvienam tėviniam mazgui:
  - Jei jo dviejų vaikų simbolių aibės turi bendrų simbolių, tai tėviniam ejų aibių daugyba.
  - Jei bendrų simbolių nėra - bus priskirta atitinkama suma - tėvinio mazgo simbolių aibė turės apjungtus abiejų vaikų simbolius.

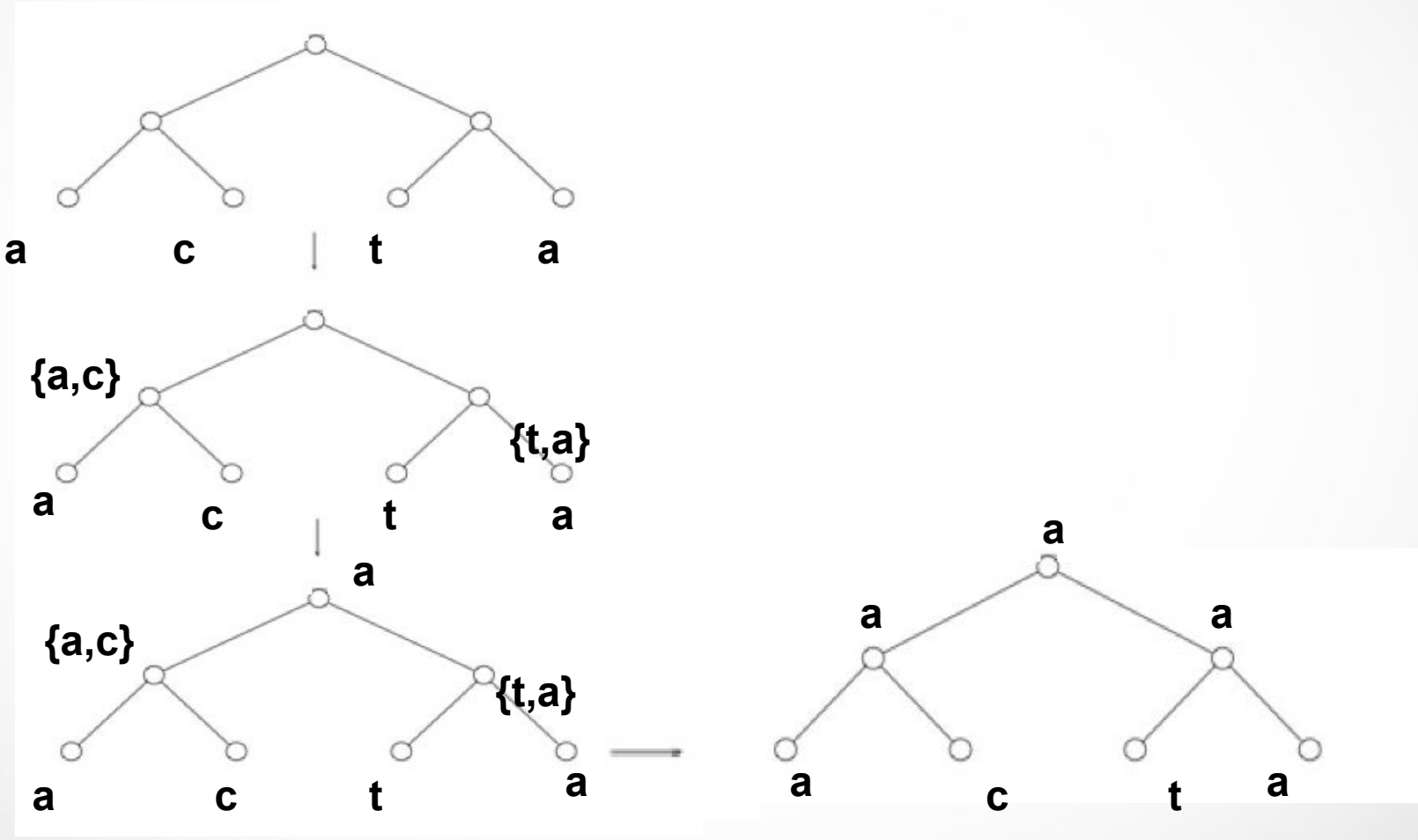
Fitch Algritmas: Pavyzdys



Fitch Algritmas

- Kitam etape priskiriam simbolį kiekvienam mazgui einant einant atgal nuo šaknies iki lapų.
  - Priskiriam šakniniam mazgui simbolį atsitiktinai iš atitinkamo jo simbolių rinkinio.
  - Kitiems mazgams, jei jo tevinio mazgo priskirtasis simbolis yra jo simbolių rinkinyje, tai jam priskiriamas jo tėvinio mazgo priskirtasis simbolis.
  - Jeigu ne - tuomet mazgui priskiriame atitiktinį simbolų iš jo simbolių rinkinio.

Fitch Algritmas: Pavyzys

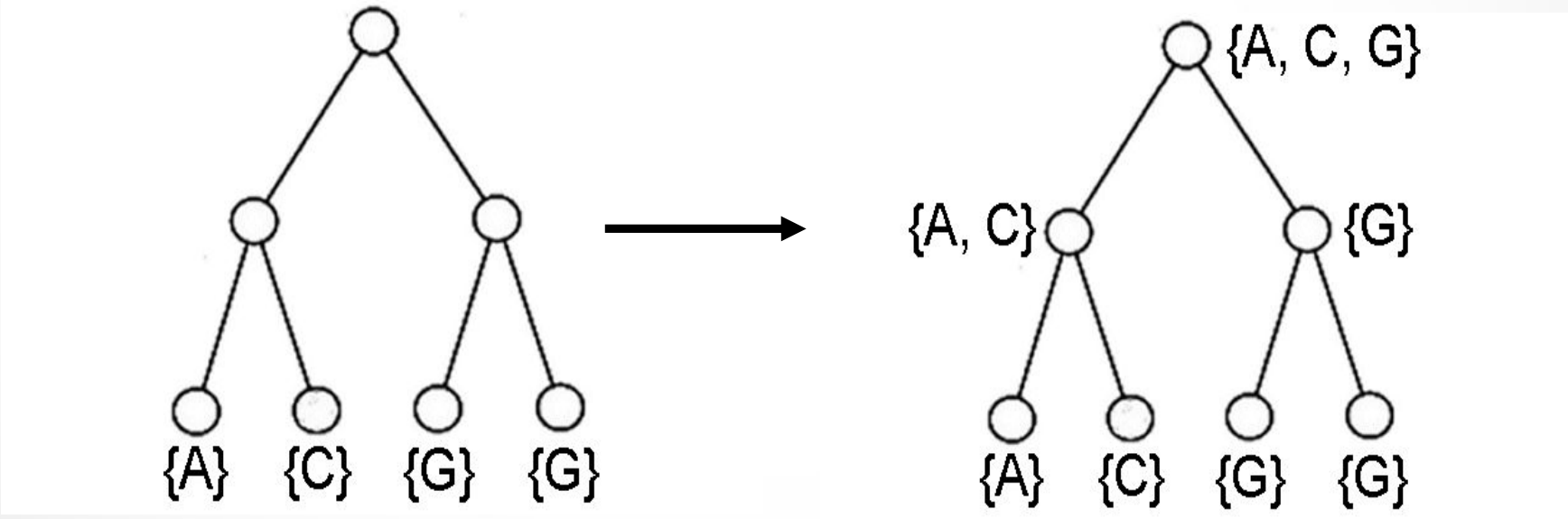


Fitch vs. Sankoff

- Abiejų veikimo laikas proporcingas O(*nk*).
- Ar jie iš esmės yra skirtingi algoritmai?
- Palyginkim?

Fitch

- Kaip matėme anksčiau:



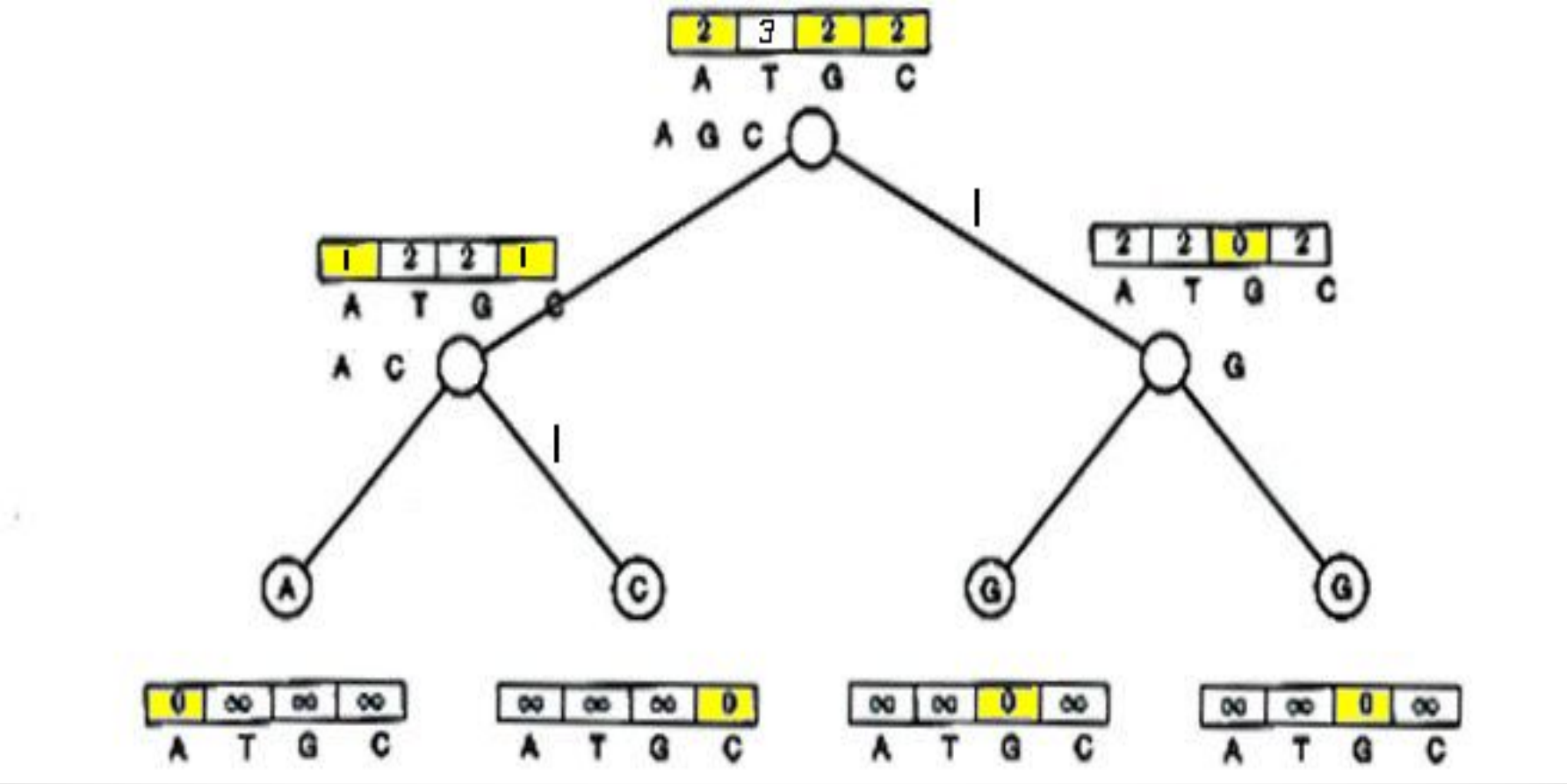
Fitch ir Sankoff palygnimas

- Įverčių matrica Fitch algoritmo atveju yra tiesiog:

	A	T	G	C
A	0	1	1	1
T	1	0	1	1
G	1	1	0	1
C	1	1	1	0

- Išsprendžiam analogišką uždavinį, kurį sprendėme naudodami Fitch algoritmą, naudodami Sankoff algoritmu.

Sankoff


2 3 2 2
A T G C
A G C
1
1 2 2 1
A T G C
A C
1
2 2 0 2
A T G C
G
A
C
G
G
0 ∞ ∞ ∞
A T G C
∞ ∞ ∞ 0
A T G C
∞ ∞ 0 ∞
A T G C
∞ ∞ 0 ∞
A T G C

Sankoff vs. Fitch

- Sankoff algoritmo atveju, simbolis *t* yra optimalus mazgui *v, jei* $s_t(v) = \min_{1<i<k} s_i(v)$.
  - Tegul optimalių simbolių rinkys mazgui *v* žymimas kaip *S*(*v*).
    - Jei *S*(kairysis vaikas) ir *S*(dešinysis vaikas) persidengia, *S*(tėvas) yra sandauga..
    - Priešingai - yra suma S(kairysis vaikas) ir S(dešinysis vaikas). Tai yra tas pats, kaip ir Fitch rekursija.
  - Taigi šie du algoritmai yra identiški.

Didelė taupumo problema

Didelė taupumo problema

- Įvestis: *n* x *m* matrica *M* aprašanti *n* rūšių, kurių kiekviena atstovaujama *m*-simbolių eilučių.

- Rezultatas:medis *T* su *n* lapų bei vidiniais mazgais su jiems priskirtomis sekomis, tokiomis kad bendras medžio taupumo įvertis būtų mažiausias.

Didelė taupumo problema

- Galima paieškų erdvė yra milžiniška.
- n lapų šakninio medžio galimų variantų skaičius:

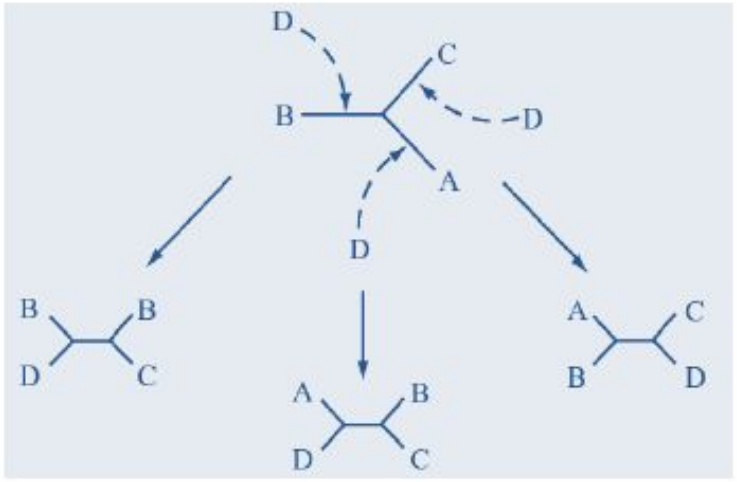
$$\frac{(2n-3)!}{2^{n-2}(n-2)!}$$

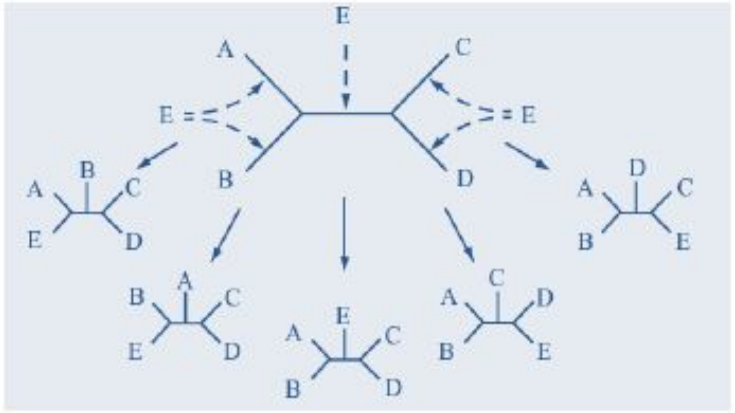
- Nudojant tiesioginį perrinkimą visų galimų medžių ieškant taupiausio "brute force" laikas proporcingas **$O(Nk^N)$**, kur
  k - galimų simbolių skaičius (DNR atveju 4)
  n - lapų skaičius

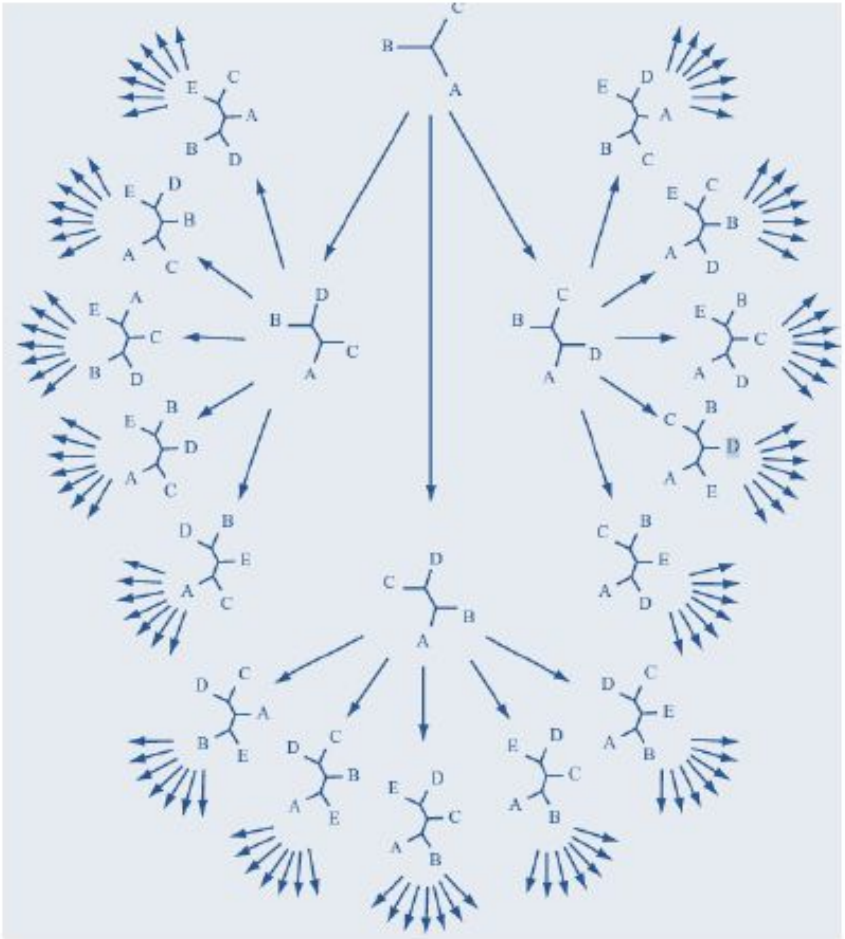
- Taigi taupumo problema yra (ang.) N*P*-Complete.
  - aptarti algoritmai realioms sekoms galima taikyti tik esant labai **mažiems n (< 10).**

- Taigi "šakok ir rišk" metodai bei euristiniai algoritai yra naudojami.

"Brute force", 6 rūšių atvejis



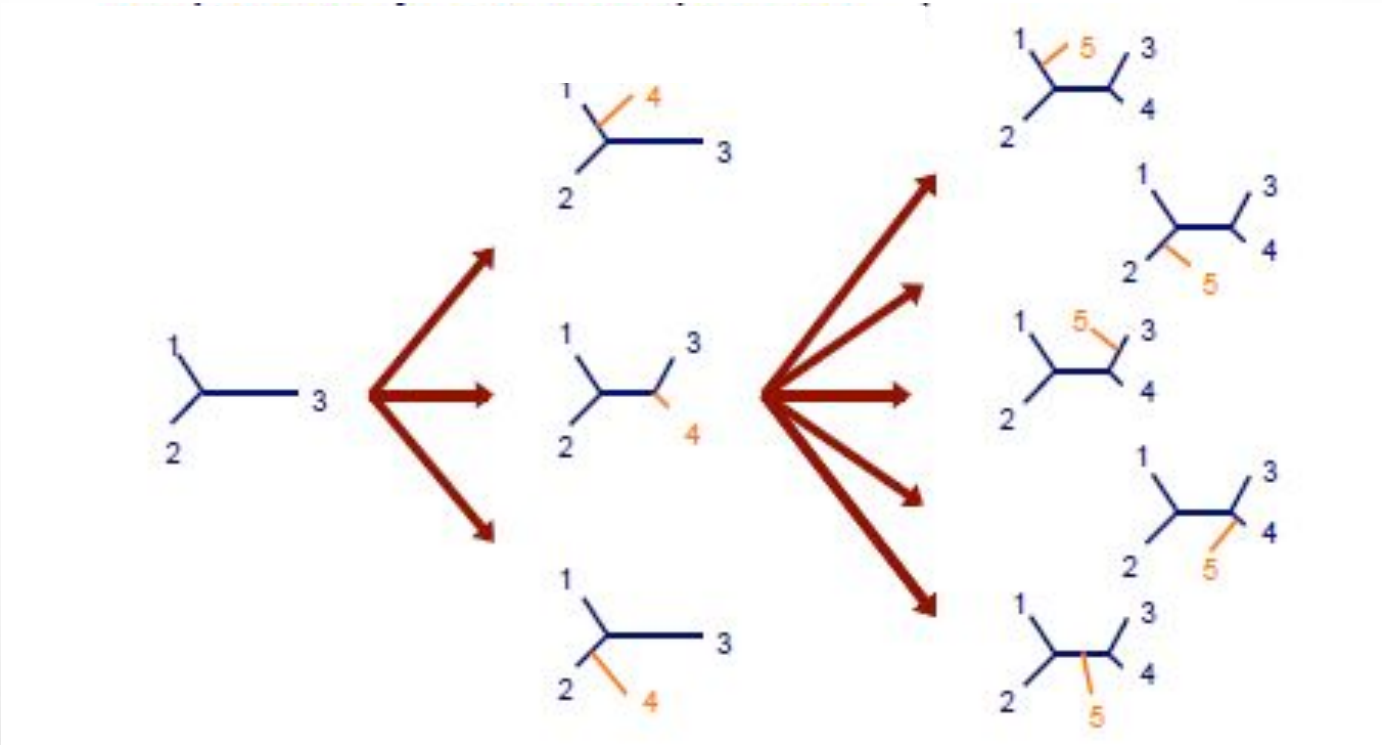




Šakok ir rišk metodas

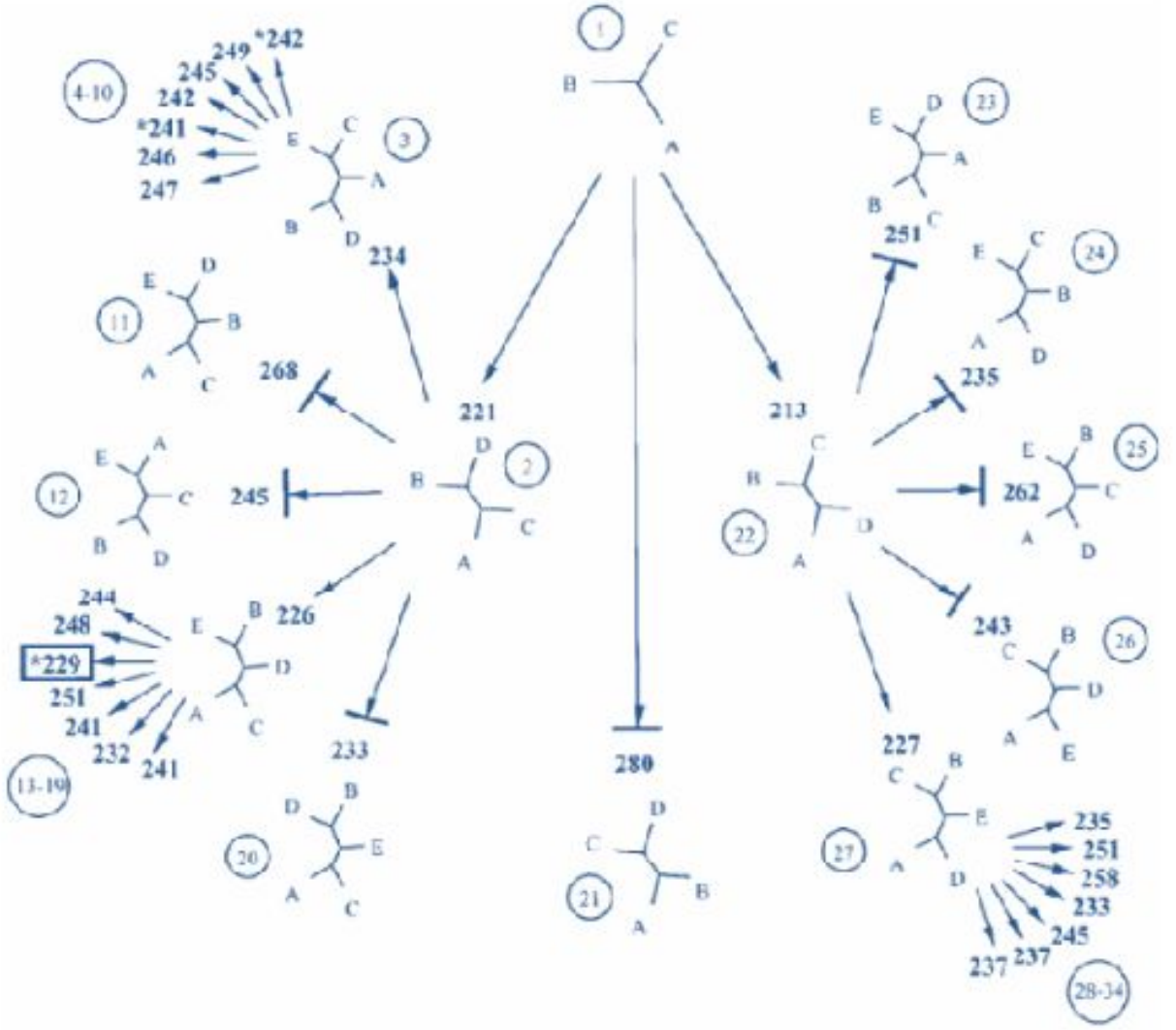
Šakok ir rišk

- Kiekvienas dalinis medis atitinka rinkinį pilnų medžių
- Dalinio medžio taupumo įvertis atitinka mažiausią taupumo įvertį, kurį galėtų turėti atitinkami pilni medžiai.
- Šakojame medį paeiliui pasirinkdami mažiausio taupumo įverčio dalinius medžius.



Šakok ir rišk. 6 sekų pavyzdys

Apskritimai - žingsnio nr. Skaičiai - taupumo įverčiai.

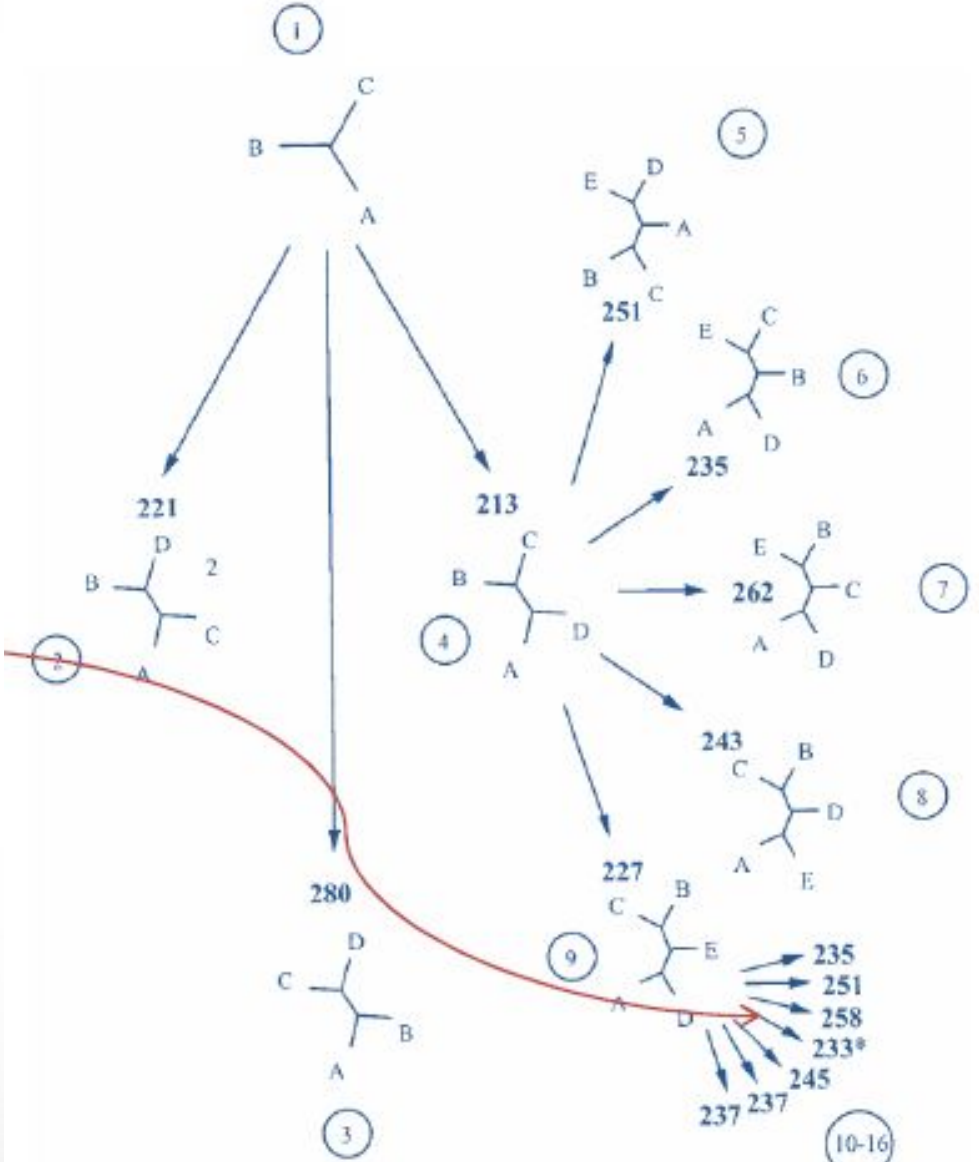


Euristinis algoritmas: Pridėjimas paeiliui "Godusis algoritmas"

Euristinis algoritmas:Pridėjimas paeiliui 6 sekų pavyzdys

- Pradeda nuo trijų sekų
- Paeiliui prideda seką, mažiausiai didinančią taupumo įvertį

- Linkęs užstrigti lokaliame minimume.



Euristinis algoritmas: Artimiausio kaimyno sukeitimas

Artimiausio kaimyno sukeitimas

- **Artimiausio kaimyno sukeitimas**: šakų sukeitimo algoritmas.

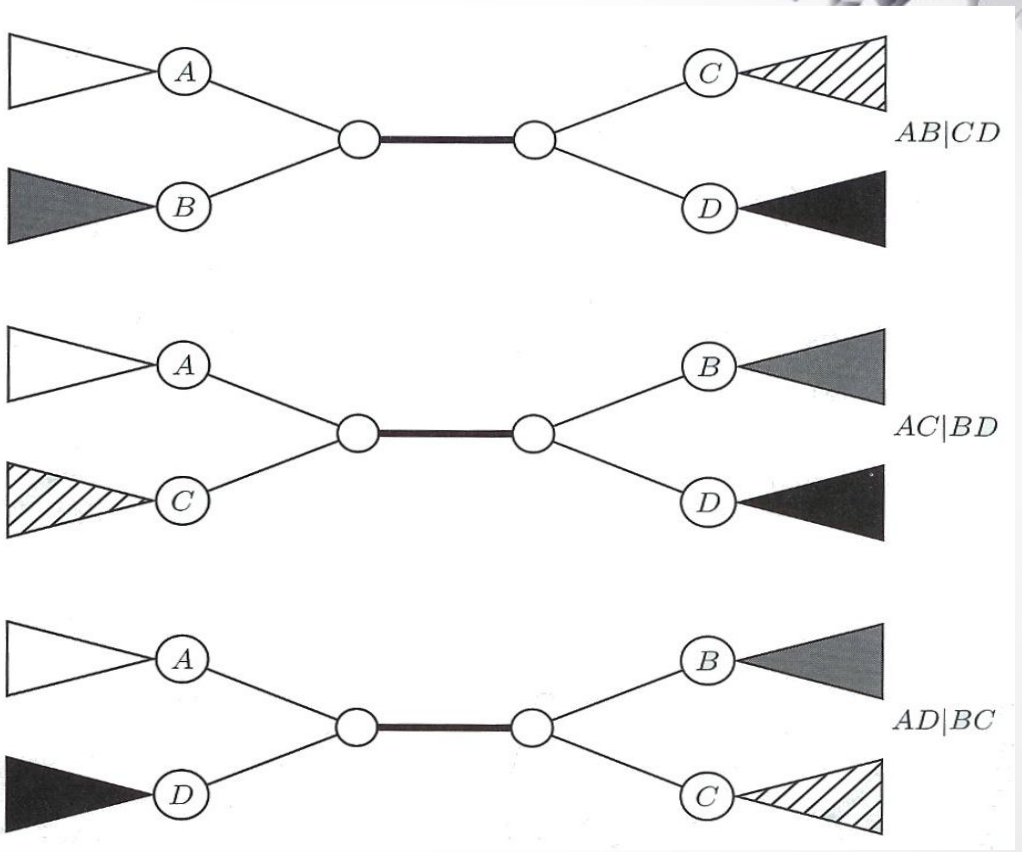
- Vertina tik dalį visų galimų šakų.

- Medžio kaimynas duotam medžiui yra tas, kuris gautas:
  - Pertvarkant keturis dalinius medžius, atsišakojančius vienos vidinės jungties tarp dviejų mazgų.Yra galimi tik trys galimi tokie pertvarkymai.

Kaimynai: Pavyzdys

- **Medžiai dešinėje yra kaimynai vienas kitam.**

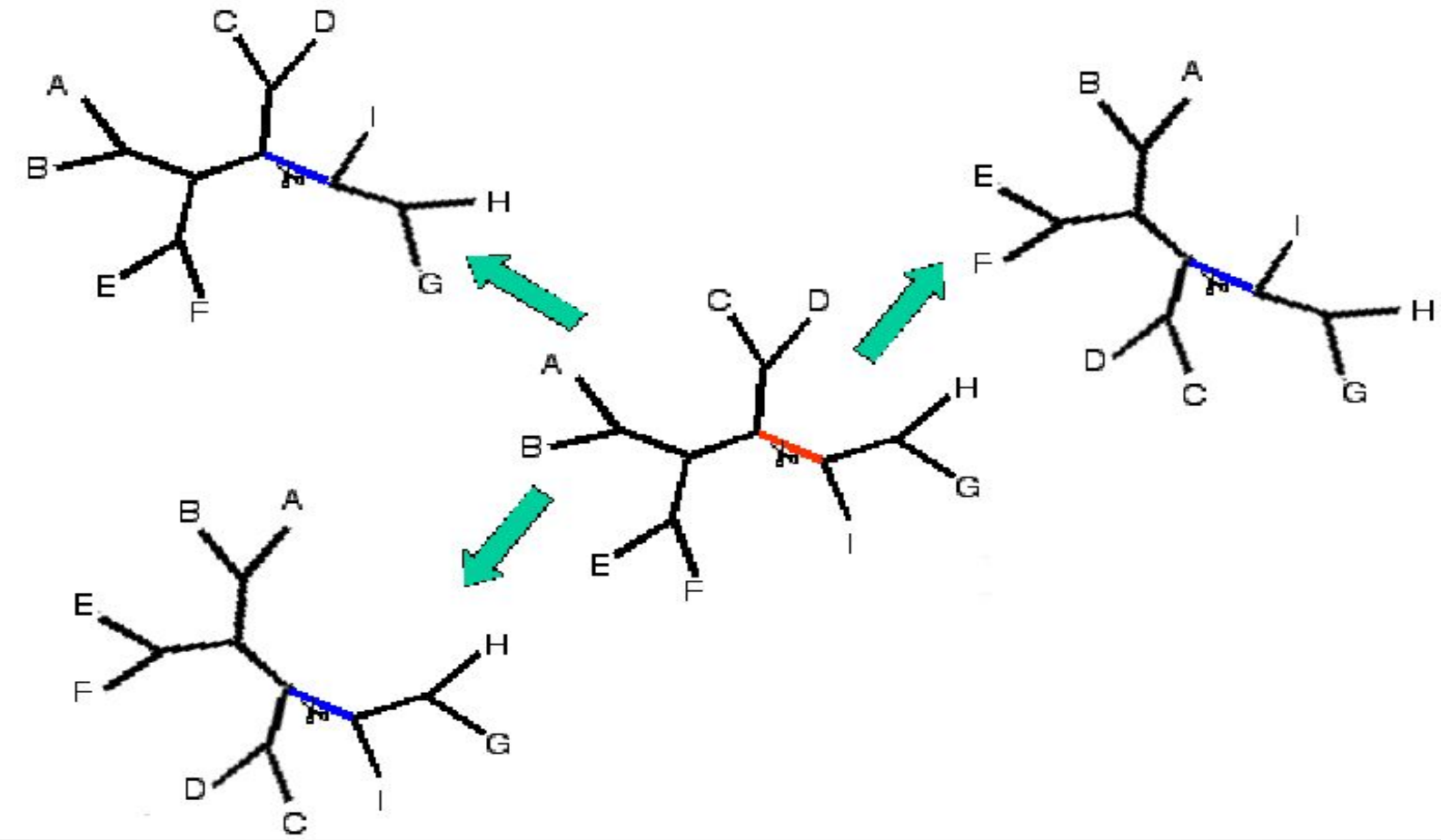
- **Struktūra žemiau A, B, C, D yra nesvarbi nes ji nekinta.**



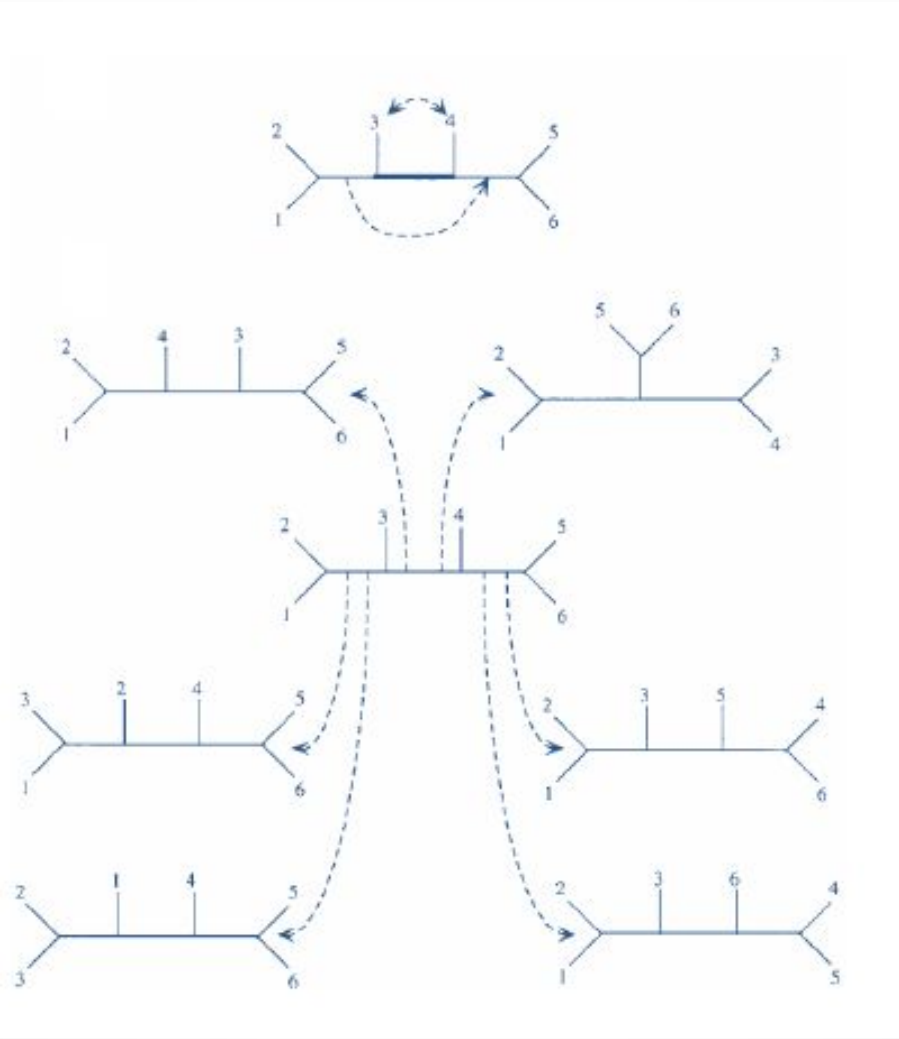
Artimiausio kaimyno sukeitimas:

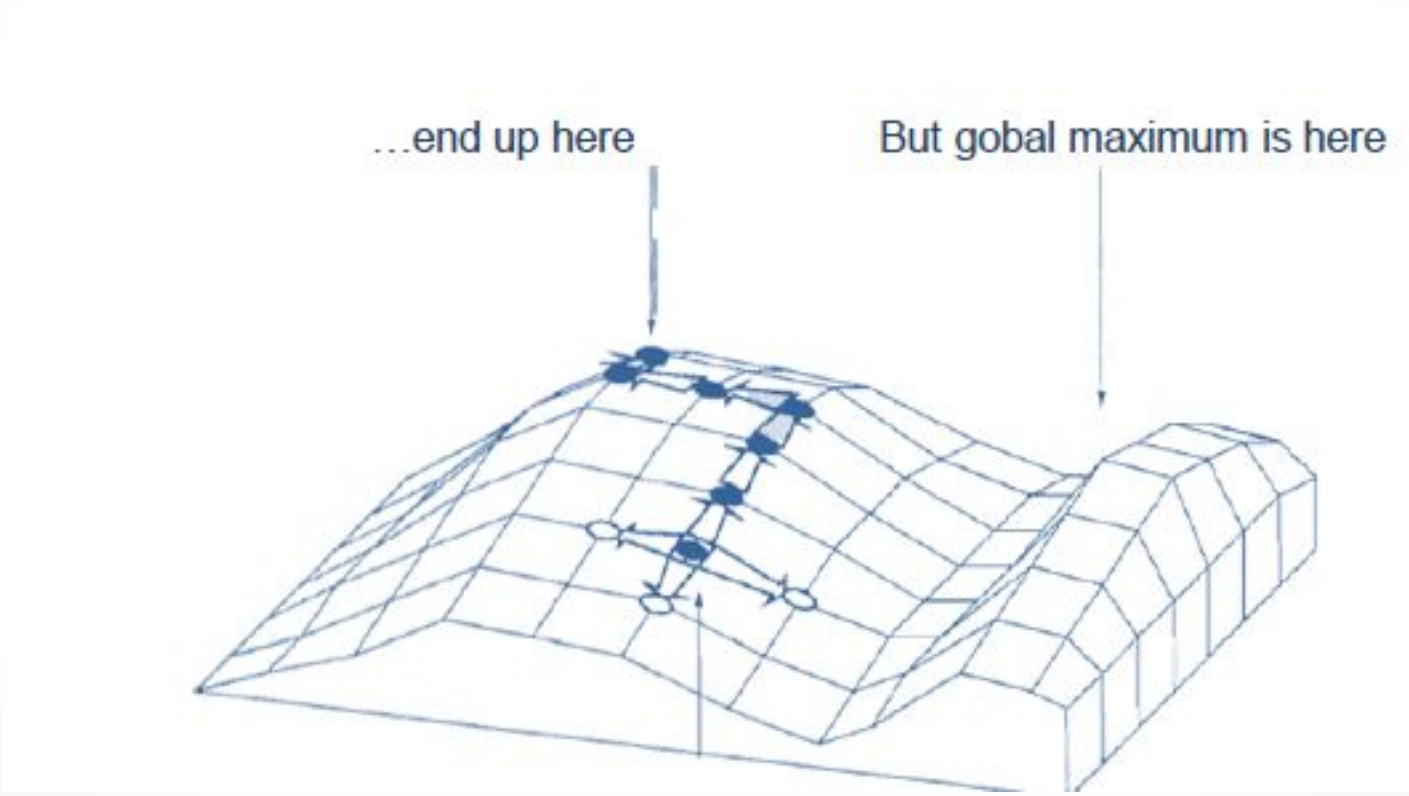
- Pradėk su atsitiktiniu medžiu, atsitiktinai parikta jungtimi ir patikrink jo kaimynus.
- Jei kaimynas turi mažesnį taupumo įvertį - pasirink jį.
- Deja nėra būdų sužinoti ar tai yra pats "taupiausias" medis.
- Lengvai galime "užstrigti" lokaliame minimume.

Artimiausio kaimyno sukeitimas: Pavyzdys



Artimiausio kaimyno sukeitimas: Pavyzdys



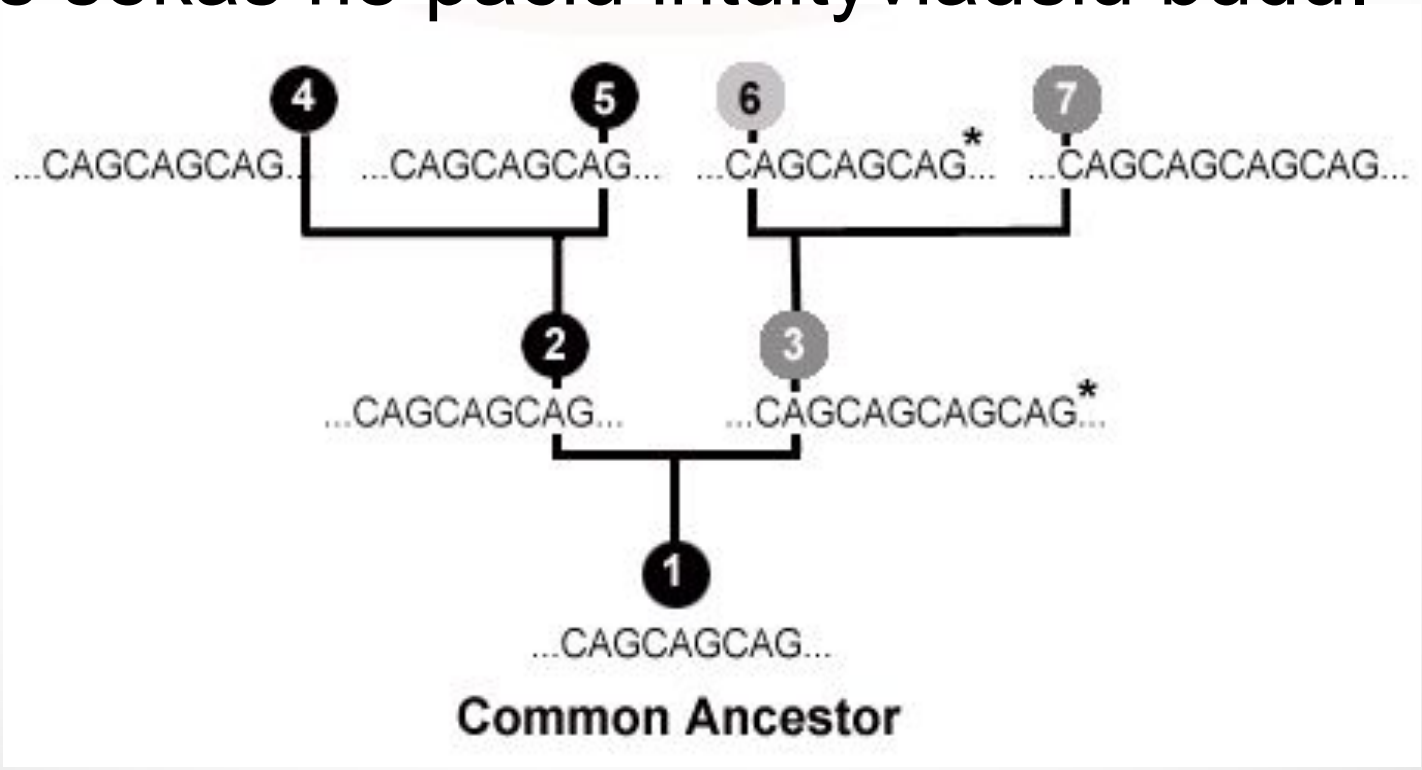

…end up here
But gobal maximum is here

Hooplazija

- Duota:
  1. CAGCAGCAG
  2. CAGCAGCAG
  3. CAGCAGCAGCAG
  4. CAGCAGCAG
  5. CAGCAGCAG
  6. CAGCAGCAG
  7. CAGCAGCAGCAG
- Dauguma grupuotų 1, 2, 4, 5, ir 6, kaip išsivysčiusią iš bendro protėvio ir kad viena mutacija sukūrė sekas 3,7.

Homoplazija

- Bet jeigu tai būtų tikras medis:
- Sikian didžiausio taupumo sugrupuotume šias sekas ne pačiu intuityviausiu būdu.



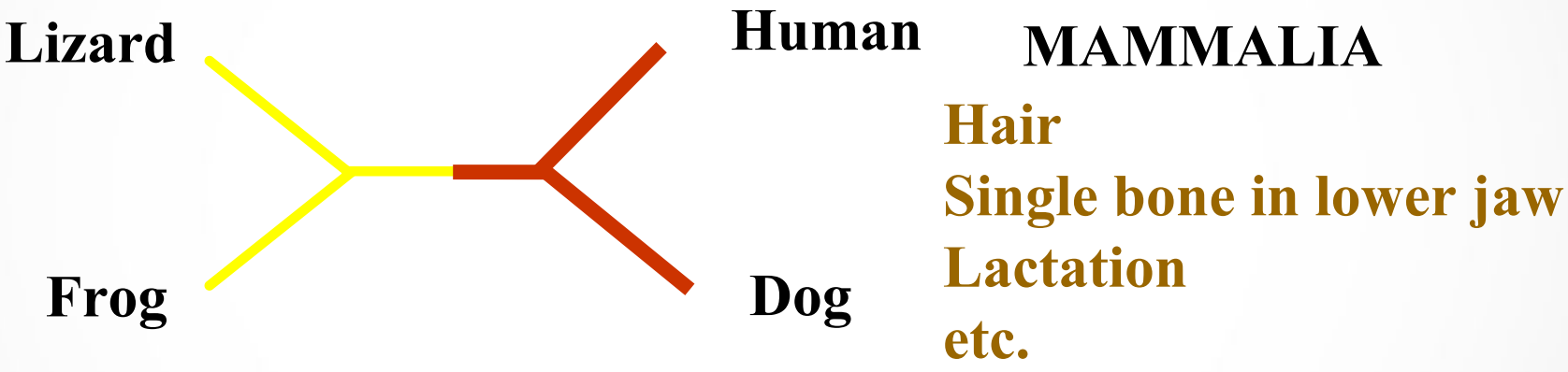
Homoplazija

- **Homoplazija**: Nepriklausomas (lygiagretus) išsivystymas panašių, identiškų savybių.

- Taupumo algoritmai mažina homoplazija, taigi jei realybėje homoplazija yra dažna - taupumo metodu gautas edis bus rezultatas bus netikslus.

Prieštaringi simboliai

- Eoliucinis medis yra tuo patikimesnis, kuo daugiau simbolių (savybių) jį remia.



- Šiuo atveju uodegos yra homoplazinis požymis. Yra dauiau įrodymų, kad šunys yra artimesni žmonėms nei varlės...labiau tikėtina,kad žmonės prarado uodegas.

How Many Times Evolution Invented Wings?

From www.bioalgorithms.info

- Whiting, et. al. (2003) looked at winged and wingless stick insects

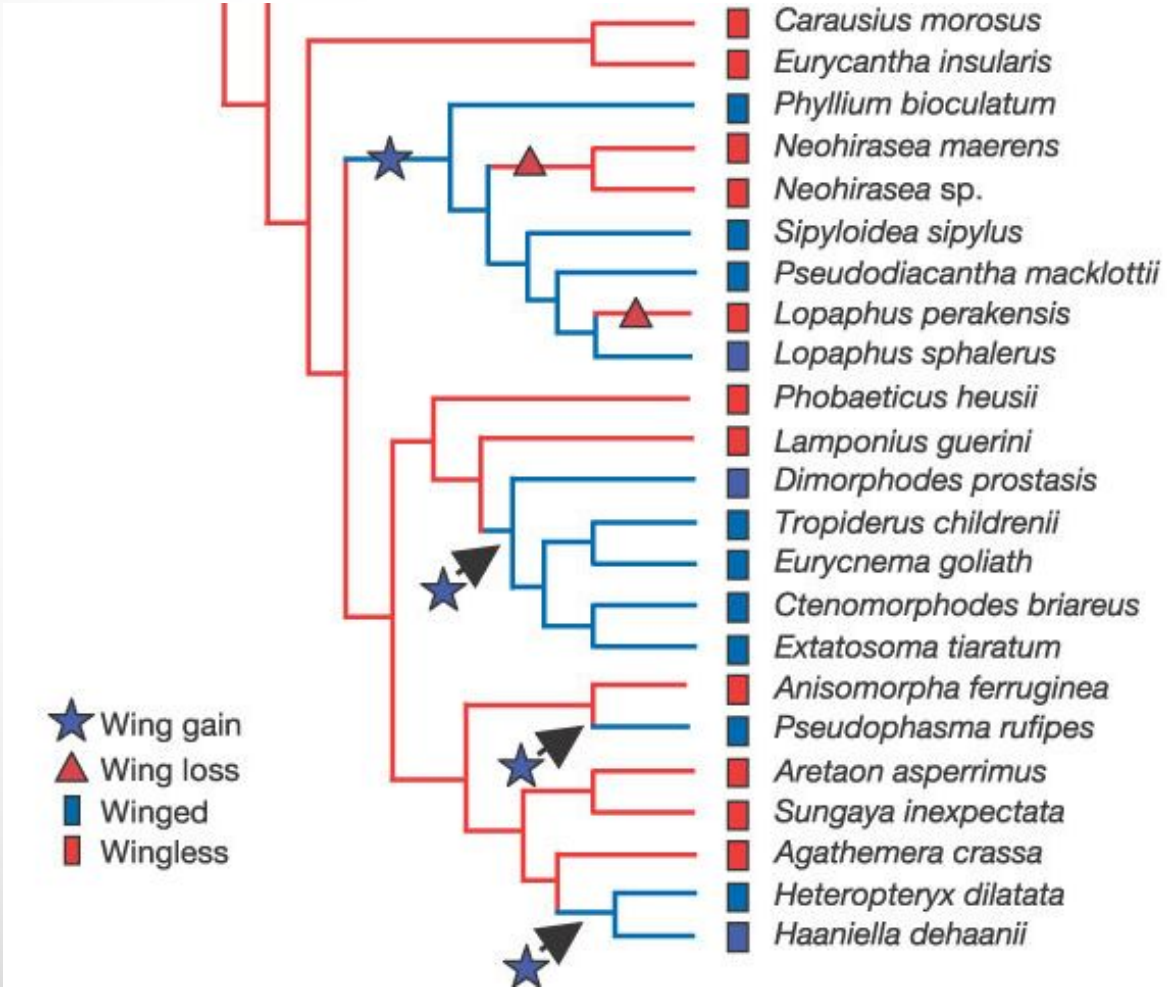
From
www.bioalgorithms.info

Reinventing Wings

- Previous studies had shown winged → wingless transitions
- Wingless → winged transition much more complicated (need to develop many new biochemical pathways)
- Used multiple tree reconstruction techniques, all of which required re-evolution of wings

Most Parsimonious Evolutionary Tree of Winged and Wingless Insects

From www.bioalgorithms.info



- **The evolutionary tree is based on *both* DNA sequences and presence/absence of wings**

- **Most parsimonious reconstruction gave a wingless ancestor**

Will Wingless Insects Fly Again?

From
www.bioalgorithms.info

- Since the most parsimonious reconstructions all required the re-invention of wings, it is most likely that wing developmental pathways are conserved in wingless stick insects

Problems with Parsimony

- It is important to keep in mind that reliance on only one method for phylogenetic analysis provides an incomplete evolutionary picture.

- When different methods (parsimony, distance-based, etc.) all give the same result, it becomes much more likely that the result obtained is in fact correct.