Fylogenetické stromy

Bioinformatika

Tomáš Martínek

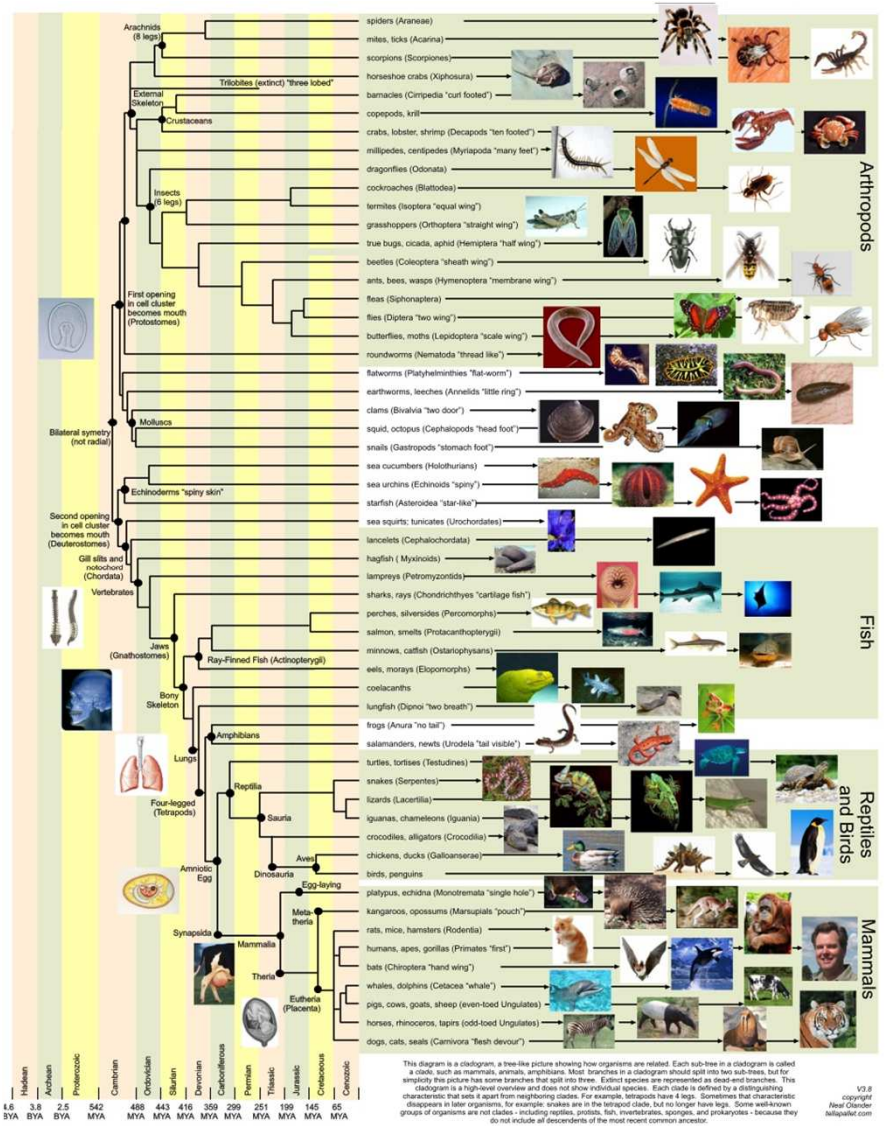
martinto@fit.vutbr.cz

Osnova

- Motivace
- Substituční modely
- Metody založené na vzdálenosti
  - Čtyřbodová podmínka
  - Neighbors-Joining
  - UPGMA
- Metody založené na znacích
  - Fitchův a Sarkoffův algoritmus
  - Nearest neighbor interchange
- Ověření korektnosti stromu
- Shrnutí

Co je to fylogenetický strom?

- Fylogenetický (evoluční) strom vyjadřuje hierarchii vývoje skupiny organizmů na základě jejich DNA
  - *Listové uzly* - reprezentují existující organizmy
  - *Vnitřní uzly* - reprezentují společné předchůdce (jejich DNA obvykle není k dispozici)
  - *Kořen* - reprezentuje evolučně nejstaršího předchůdce
  - *Délky hran* – obvykle vyjadřují dobu, kdy došlo k oddělení vývoje nebo množství změn mezi organizmy



Historie fylogenetiky

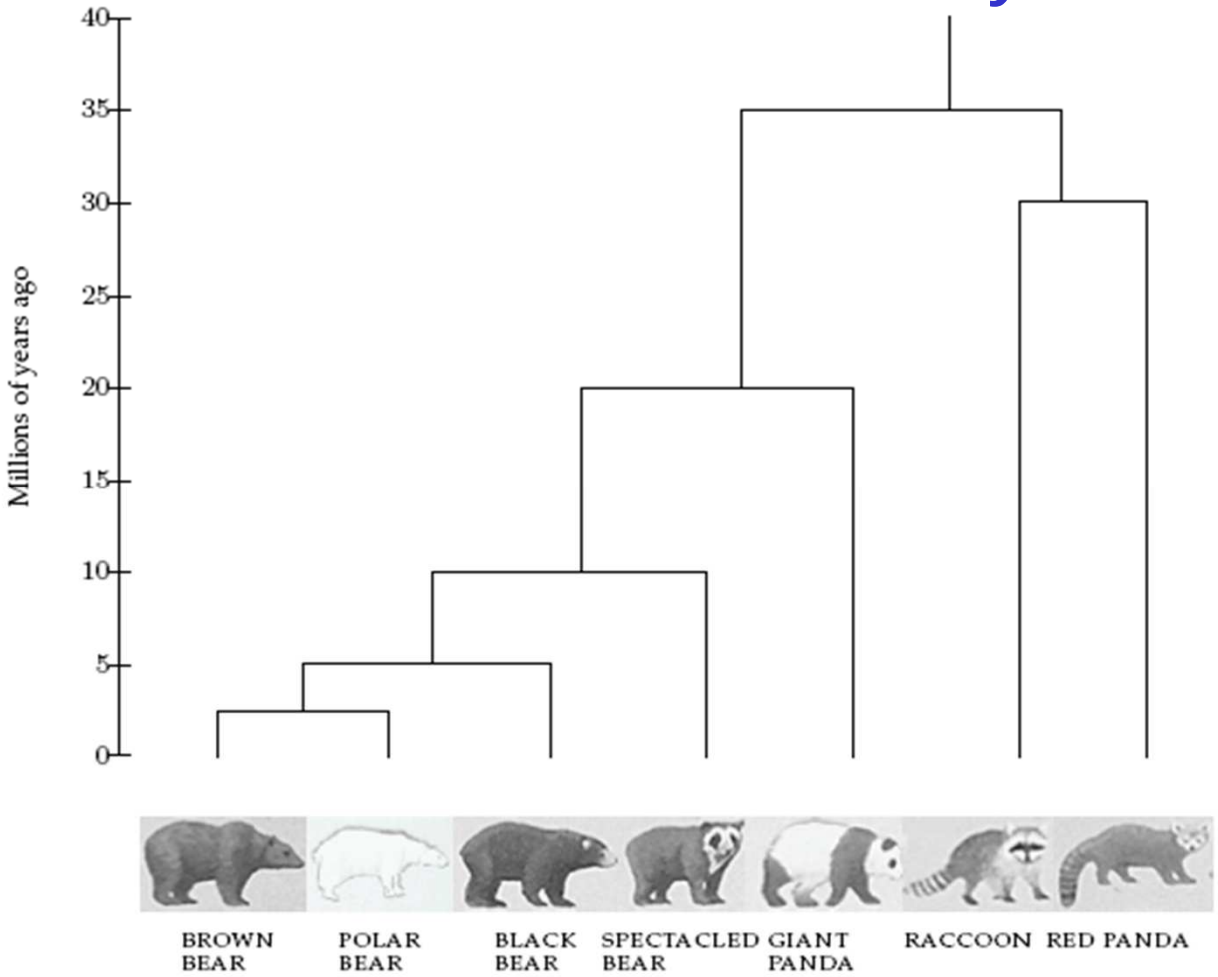
- Taxonomie - věda která se zabývá pojmenováním a seskupováním různých organizmů do skupin (klasifikace) - prováděla se ještě dříve, než se vůbec vědělo o existenci genetické informace uložené v genomech
- Před tím, než byly k dispozici genetická data, používali taxonomisté informaci o tom, jak organizmus vypadá (fenotyp)
- Problémy těchto přístupů:
  - nesprávné stromy, pokud je špatně zvoleno kritérium
  - u organizmů, které nemají zjevné charakteristiky, např. bakterie.
- G.H.F.Nuttal (1902-4) demonstroval vzdálenost organizmů na příkladě intenzity imunitní reakce (vložením krve jednoho organizmu do druhého)
- Touto technikou byly prozkoumány vztahy stovek organizmů a vznikla také poprvé myšlenka, že člověk a opice mají společného předka
- Analýza na úrovni DNA dává nejspolehlivější výsledky

Rozluštění hádanky o pandě velké

- Přibližně 100 let trvalo, než dokázali vědci rozhodnou, do které kategorie patří panda velká
- Panda velká vypadá jako medvěd, ale má určité znaky neobvyklé pro medvědy, ale typické pro mývaly, např. nemají zimní spánek
- V roce 1985 Steven O'Brien s kolegy vyřešili problém klasifikace pandy analýzou DNA sekvencí

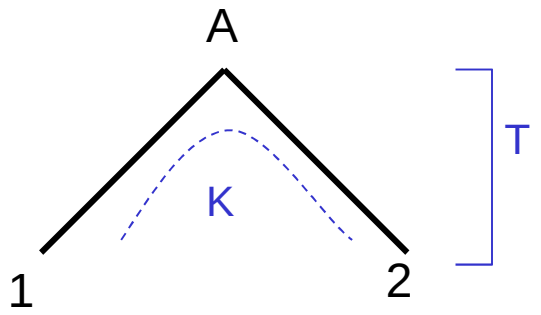
Rozluštění hádanky o pandě velké

- Evoluční strom medvědů a mývalů



Konstrukce fylogenetických stromů

- Jak určit vzdálenost dvou organizmů od jejich společného předka?
  - Obvykle jsme schopni vypočítat editační vzdálenost (počet mutací)
  - Předpokládejme jednoduchý model, kde ke změnám (mutacím) dochází v pravidelných intervalech určitou rychlostí
  - Pokud bychom znali tuto rychlost mutací, jsme schopni odvodit i vzdálenost od společného předka



- Rychlost mutací:
  - porovnají se dvě sekvence a zjistí se počet mutací - hodnota *K*
  - pokud k těmto změnám došlo za čas *T*, potom rychlost mutace lze vypočítat jako:
  - $r = K/(2T)$
  - hodnota je dělena dvěma – předpokládá se, že ke změnám došlo od společného předchůdce *A*

Mutace vs. Substituce

- Odpovídají změny, které pozorujeme v DNA skutečnému počtu mutací generovaných určitou rychlostí?
- Mutace:
  - záměna nukleotidu za jiný
  - výhodné, nevýhodné, neutrální
- Přirozený výběr
  - Proces, kdy pouze výhodné nebo neutrální mutace jsou přenášeny do replikované DNA
  - Nevýhodné mutace vedou obvykle k uhynutí organizmu
- Mutace vs. Substituce
  - *Mutace* - změny při replikaci nebo opravě DNA
  - *Substituce* - pouze ty mutace, které projdou filtrací přirozeného výběru a uplatní se

Mutace vs. Substituce

- Princip přirozeného výběru potvrzuje i různou četnost substitucí na různých částech DNA
- U částí, které jsou důležité pro zachování správné funkce organismu dochází ke změnám méně častěji
  - nejméně změn v kódující oblasti
  - více změn nastává v oblastech, které se transkribují, ale nepřekládají se
  - nejvíce změn v nekódujících oblastech
- Podobně pro vložení resp. odstranění znaku (*Indel*)
  - v kódující oblasti genu by způsobila posun celého rámce pro překlad na protein
  - taková změna je příliš velká, příslušné enzymy toto hlídají a odstraňují v souladu z přirozeným výběrem
  - výskyt indelu je proto zhruba 10 krát menší v kódujících oblastech, než v nekódujících

Mutace vs. Substituce

- *V kódujících oblastech rozlišujeme synonymní vs. nesynonymní substituce*
  - jednotlivé aminokyseliny skládají z kodonů
  - tvoří se ale pouze 20 aminokyselin z 64 možných kombinací, přičemž některé kombinace kodonů generují stejnou aminokyselinu
  - mutace v kódující oblasti, které nezpůsobí změnu výsledné aminokyseliny se nazývají *synonymní*
  - v opačném případě se nazývají *nesynonymní*
  - dle předpokladu výskyt synonymních substitucí je daleko častější než nesynonymních (viz. tabulka)

Mutace vs. Substituce

- Tabulka: poměr synonymních změn na synonymní místo (K_S) a počet nesynonymních změn na nesynonymní místo (K_N) pro různé geny savců

Gen	**#Kodonů**	**Člověk/myš**		**Člověk/kráva**		**Člověk/králík**	
	(člověk)	**K_S**	**K_N**	**K_S**	**K_N**	**K_S**	**K_N**
Erythropoietin	194	0,481	0,063	0,242	0,068	0,394	0,070
Prolactin	226	0,364	0,098	0,368	0,085	0,395	0,064
Alpha globin	143	0,584	0,022	0,236	0,025	0,204	0,038
Beta globin	148	0,324	0,033	0,271	0,046	0,294	0,015
Histone 2A	115	0,967	0,057	1,110	0,057	0,174	0,034

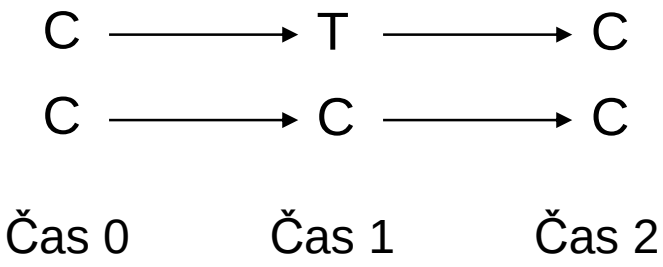
- Kuriozitou je, že pro některé geny je četnost nesyn. substitucí dokonce větší než těch synon. - rodina genů HLA (human leukocyte antigen), sloužící pro identifikaci cizích antigenů v rámci imunitního systému (K_S=3,5% a K_N=13,3%)
- Přirozený výběr si vynucuje rychlé změny, aby byl organizmus schopen lépe reagovat na nové viry

Mutace vs. Substituce

- Zjistit frekvenci mutací je složité díky filtraci prováděné přirozeným výběrem
- Pro přibližné určení těchto frekvencí v rámci kódujících oblastí se používají synonymní a nesynonymní substituce
  - *Synonymní* - nejvíce se podobají frekvenci mutací
  - *Nesynonymní* - až po průchodu přirozeného výběru
- Důsledky:
  - Při konstrukci stromu je nezbytné vybrat vhodné části genomu
  - Pokud jsou vybrány kódující oblasti (geny), potom počet synonymních substitucí se nejvíce podobá skutečné rychlosti mutací
  - Poměr mezi synonymními a nesynonymními substitucemi navíc udává jak je daná část důležitá z pohledu funkce

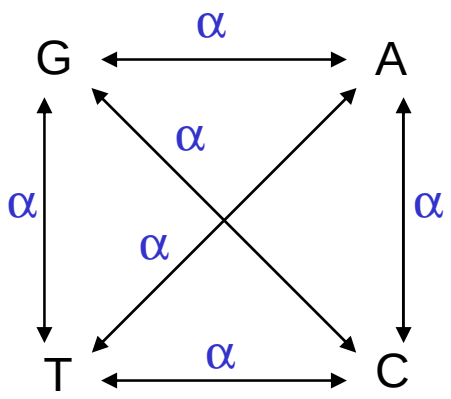
Určení počtu substitucí

- Odpovídají změny, které pozorujeme v DNA skutečnému počtu substitucí mezi dvojicí sekvencí?
  - Pokud spočítáme počet změn mezi dvojicí sekvencí, potom toto číslo nemusí odpovídat skutečnému počtu substitucí *K* - nejsou započítány situace, kdy se vlivem mutací vrátíme k nukleotidu, který byl v sekvenci na začátku!



Jukesův-Cantorův model (1969)

- Jednoduchý model, který uvažuje stejnou pravděpodobnost pro změnu z jednoho nukleotidu do druhého a případně nazpět
- Odvození vzorců:
  - Pravděpodobnost, že původní znak *C* zůstane v čase 1 na stejné pozici:
    - $P_{CC(1)} = 1-3\alpha$
  - Pravděpodobnost, že původní znak *C* zůstane v čase *t* na stejné pozici:
    - $P_{CC(t)}=1/4+(3/4)e^{-\alpha t}$
  - Skutečný počet substitucí mezi dvojicí sekvencí:
    - $K=-3/4 \ln[1-(4/3)(p)]$
    - kde *p* je počet párových změn

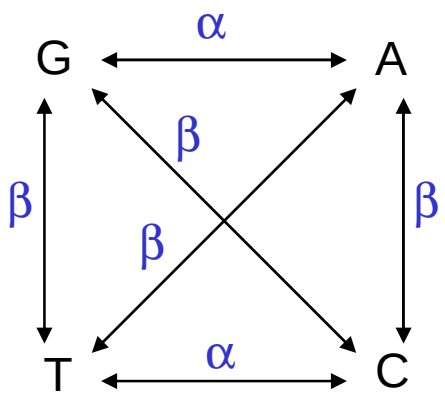


- Výsledný vzorec odpovídá předpokladům:
  1. Pokud se sekvence mění málo, potom je i malá hodnota *K*
  2. Pokud se sekvence mění hodně, potom je hodnota *K* větší než pouze počet změn mezi sekvencemi

V 70. letech se podařilo nasekvenovat první sekvence a ukázalo se, že Jukes-Cantor model je příliš zjednodušený

Kimurův model (1980)

- Puriny vs. Pyrimidiny
  - G,A = puriny
  - C,T = pyrimidiny
  - Pravděpodobnost přechodu z purinu na purin nebo pyrimidinu na pyrimidin (přechod) je 3x větší než přechod z purinu na pyrimidin nebo naopak (transverze) - dáno chemickou strukturou
- Model proto uvažuje odlišnou pravděpodobnost pro přechod mezi puriny a pyrimidiny – parametry α,β
- Odvození vzorců:
  - Pravděpodobnost, že původní znak *C* zůstane v čase *t* na stejné pozici:
    - $P_{CC(t)}=1/4+(1/4)e^{-4\beta t}+(1/2)e^{-2(\alpha+\beta)t}$



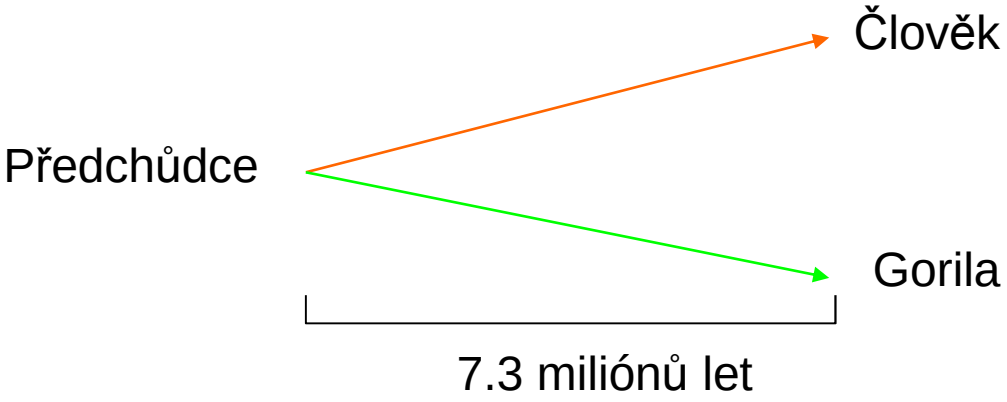
- Skutečný počet substitucí mezi dvojicí sekvencí:
  - $K= \frac{1}{2} \ln[1/(1-2P-Q)]+ \frac{1}{4} \ln[1/(1-2Q)]$
  - kde P je počet přechodů a Q je počet transverzí

Kimurův model (1980)

- V 80. letech se podařilo získat daleko více sekvencí a ukázalo se, že Kimurův model je rovněž příliš zjednodušený
- V obecném případě je potřeba uvažovat až 12 možných přechodů mezi nukleotidy a každému přiřadit svou vlastní pravděpodobnost
- Do modelu mohou být zakomponovány další zpřesňující parametry, avšak složitost matematického modelu se výrazně zvyšuje
- Důsledek:
  - Skutečný počet substitucí lze získat až po aplikaci některého ze substitučních modelů

Molekulární hodiny

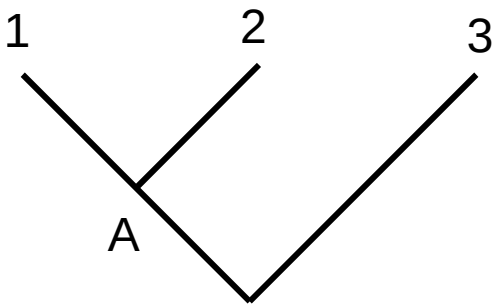
- Je rychlost substitucí skutečně konstantní?
  - Emile Zuckerkandl a Linus Pauling (1965) vytvořili první studii porovnávající rychlost změn mezi různými proteiny
  - Stavěli na předpokladu že rychlost substituce je konstantní pro daný typ proteinu (mezi proteiny je různá)
  - Zavedli tzv. molekulární hodiny (každý tik znamená určité množství změn aminokyselin)
  - Tento model by umožňoval nejen určit fylogenetický vztahy mezi organizmy, ale také dobu, kdy se jejich vývoj oddělil



Bohužel tato hypotéza hodin tikajících stále stejnou rychlostí byla vyvrácena.

Relativní test rychlosti

- Sarich a Wilson (1973) vytvořili jednoduchý test pro odhad rychlosti substituce nezávislý na znalosti času rozdělení
- Předpokládejme následující strom, kde 1 a 2 jsou příbuzné organizmy se společným předkem A, zatímco 3 je vzdálenější organizmus
- Pokud je předpoklad konstantních molekulárních hodin správný, potom platí:
  - $D_{13}=D_{A1}+D_{A3}$
  - $D_{23}=D_{A2}+D_{A3}$
  - $D_{12}=D_{A1}+D_{A2}$
- Odtud lze odvodit:
  - $D_{A1}=(D_{12}+D_{13}-D_{23})/2$
  - $D_{A2}=(D_{12}+D_{23}-D_{13})/2$
- Test je splněn, pokud jsou hodnoty D_{A1} a D_{A2} shodné

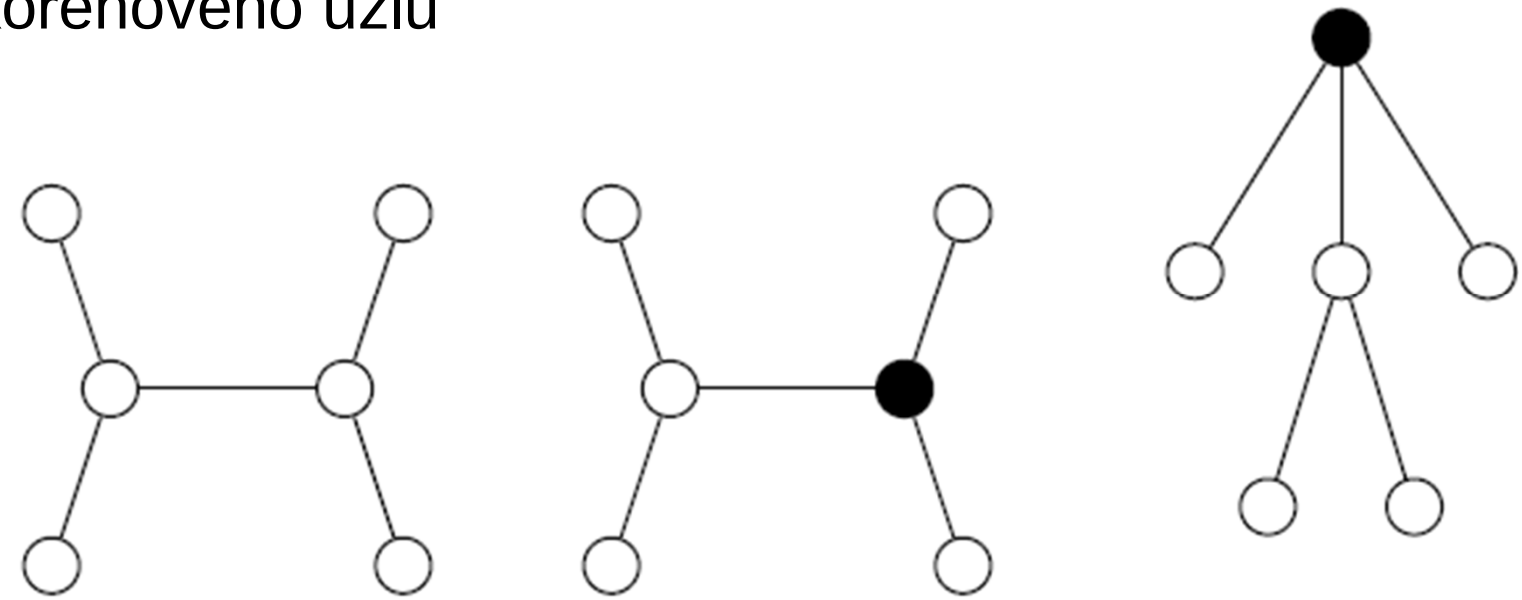


Relativní test rychlosti

- Provedené testy:
  - myš vs. krysa – test byl splněn
  - člověk vs. opice – rychlost člověka je poloviční
  - člověk vs. myš – myši vykazují dvounásobnou rychlost a odhadovaný společný předek se vyskytoval před 80 až 100 mil. lety
- Faktory způsobující různou rychlost:
  - Doba jedné generace
    - opice žijí dvakrát méně let než lidé
    - hlodavci žijí mnohem kratší dobu než opice
  - Schopnost reprodukce
  - Rychlost metabolizmu
  - Nutnost přizpůsobit se prostředí
- Důsledek:
  - výpočetní metody pro hledání fylogenetických stromů musí uvažovat různou rychlost substitucí v oddělených větvích

Evoluční stromy

- Kořenové vs. nekořenové
  - *Kořenové* – všechny analyzované organizmy mají společného předka
  - *Nekořenové* – společný předek není znám
  - Kořenový strom lze z nekořenového získat pouze identifikací kořenového uzlu

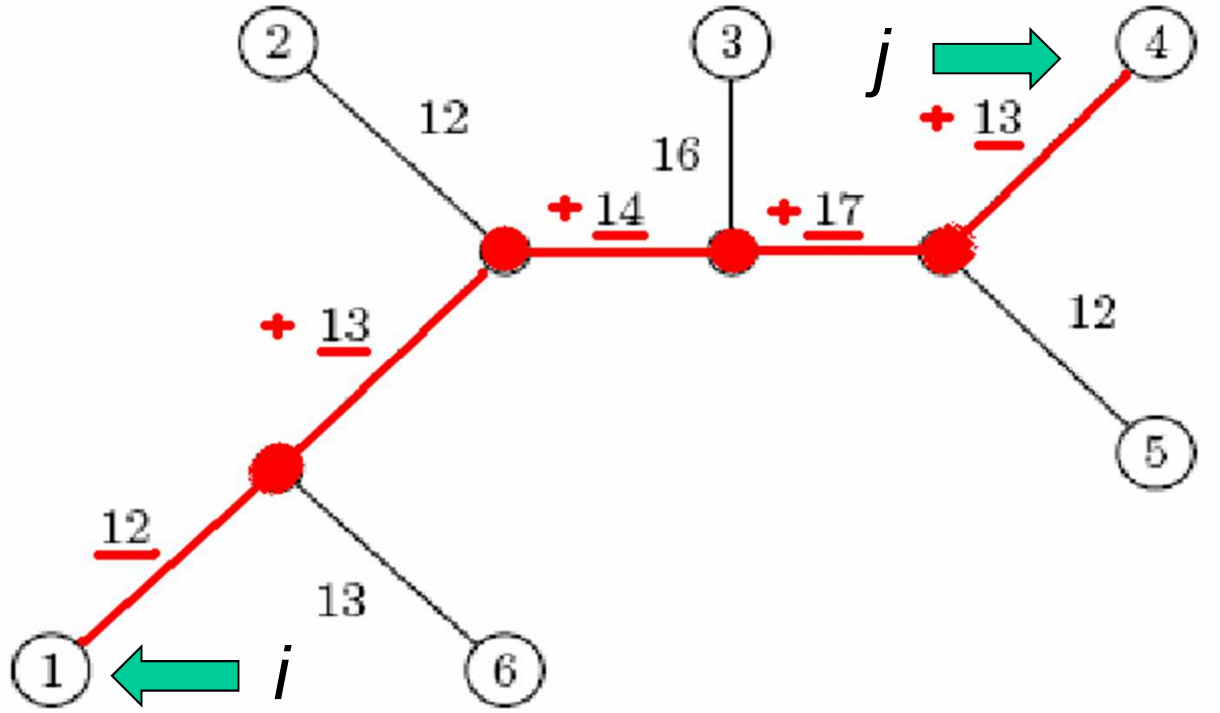


Evoluční stromy

- Ohodnocení/délka hran stromu reflektuje:
  1. Počet mutací na evoluční cestě z jednoho organizmu ke druhému
  2. Čas vývoje od jednoho organizmu ke druhému
- Při konstrukci stromu *T* nás obvykle zajímá 1. kritérium, tj. zda vzdálenost na cestě mezi uzly *i* a *j* odpovídá počtu mutací mezi organizmy
- Označujeme:
  - $d_{ij}(T)$ – stromová vzdálenost mezi uzly *i* a *j*

Evoluční stromy

- Příklad:



$$d_{1,4} = 12 + 13 + 14 + 17 + 13 = 69$$

Evoluční stromy

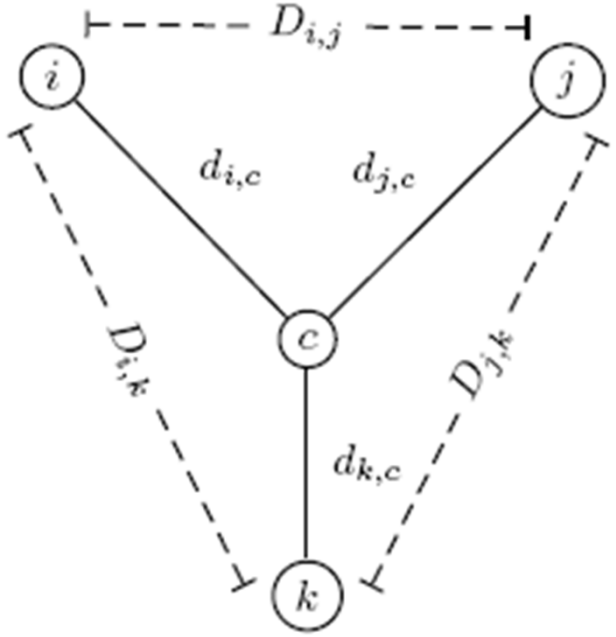
- Skutečný počet substitucí mezi jednotlivými organizmy lze spočítat na základě editační vzdálenosti D_{ij}
- Definice problému:
  - Máme k dispozici matici $n \times n$ editačních vzdáleností D_{ij}
  - Jak sestrojit evoluční strom tak, aby stromová vzdálenost odpovídala editační vzdálenosti?

Stromová vzdálenost (***neznáma***)

$$\underbrace{D_{ij}}_{\text{Editační vzdálenost (}\textbf{\textit{známa}}\text{)}} = \overbrace{d_{ij}(T)}$$

Sestrojení stromu se třemi uzly

- Řešení pro strom se třemi uzly lze snadno odvodit
- Uvažujme následující obecné schéma stromu se třemi listovými uzly a jedním centrálním uzlem:



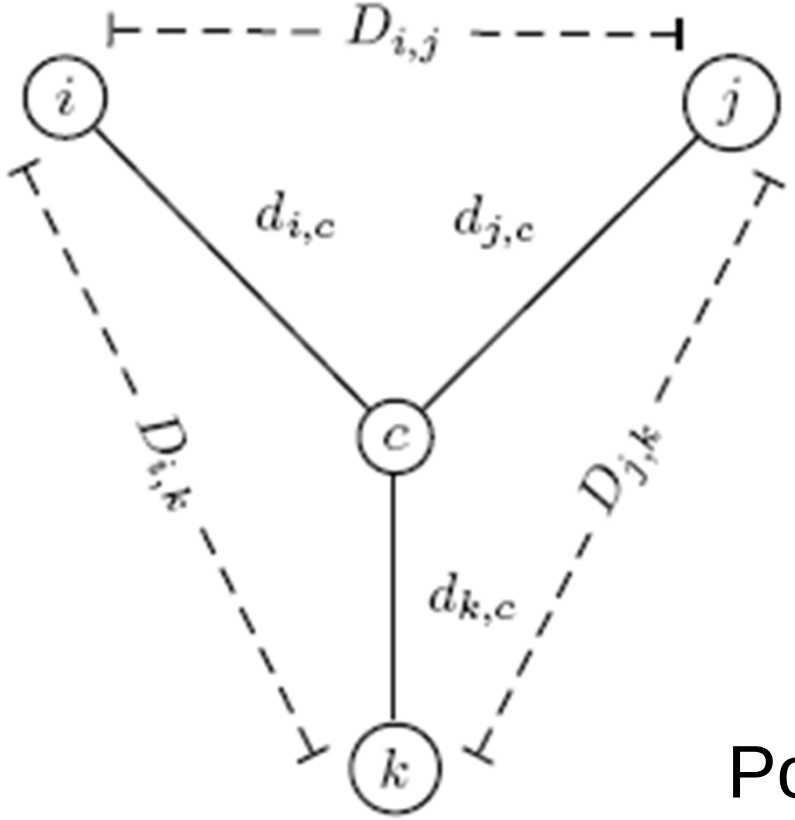
Dle grafu platí:

$$d_{ic} + d_{jc} = D_{ij}$$

$$d_{ic} + d_{kc} = D_{ik}$$

$$d_{jc} + d_{kc} = D_{jk}$$

Sestrojení stromu se třemi uzly



$$d_{ic} + d_{jc} = D_{ij}$$

$$+\ d_{ic} + d_{kc} = D_{ik}$$

$$2d_{ic} + d_{jc} + d_{kc} = D_{ij} + D_{ik}$$

$$2d_{ic} + D_{jk} = D_{ij} + D_{ik}$$

$$\boldsymbol{d_{ic} = (D_{ij} + D_{ik} - D_{jk})/2}$$

Podobně,

$$\boldsymbol{d_{jc} = (D_{ij} + D_{jk} - D_{ik})/2}$$

$$\boldsymbol{d_{kc} = (D_{ki} + D_{kj} - D_{ij})/2}$$

Sestrojení stromu s >3 uzly

- Strom s *n* listovými uzly má *2n-3* hran

- To znamená, že pro strom s *n* uzly je potřeba vyřešit soustavu $\binom{n}{2}$ lineárních rovnic s *2n-3* proměnnými

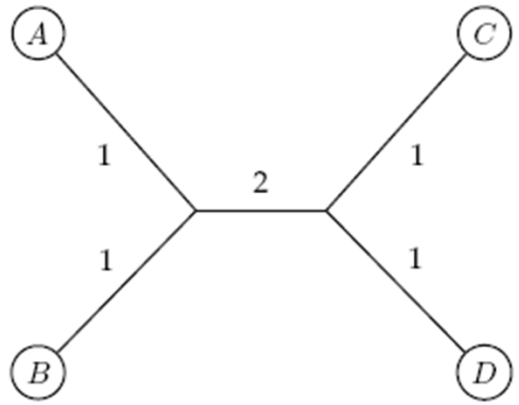
=> pro větší n je výpočet příliš složitý

=> ne vždy existuje řešení pro takový sytém rovnic

Aditivní vs. neaditivní matice

- Matice se nazývá aditivní, pokud existuje strom, pro který platí $d_{ij}(T) = D_{ij}$

	A	B	C	D
A	0	2	4	4
B	2	0	4	4
C	4	4	0	2
D	4	4	2	0



Příklad aditivní matice

- Ne vždy je možné aditivní strom sestavit z matice sestavit, potom se matice nazývá neaditivní

	A	B	C	D
A	0	2	2	2
B	2	0	3	2
C	2	3	0	2
D	2	2	2	0

?

Příklad neaditivní matice

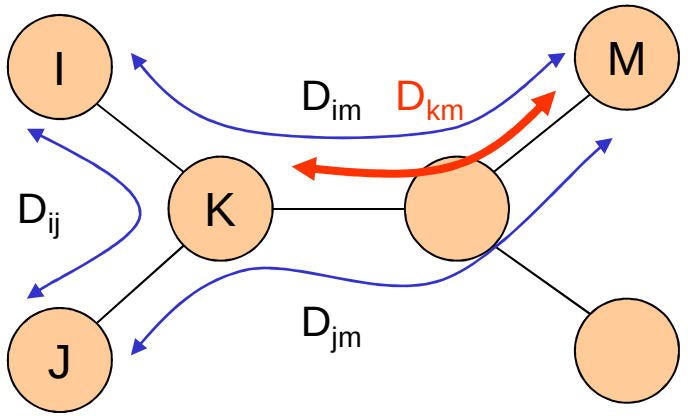
Obecná definice problému

- Cíl:
  - Sestavit evoluční strom z matice vzdáleností
- Vstup:
  - *n x n* matice vzdáleností D_{ij}
- Výstup:
  - Váhovaný strom *T* s n listovými uzly splňující vzdálenosti *D*
- Řešení:
  - Pokud je matice aditivní, potom existuje jednoduchý algoritmus pro řešení problému

Pro aditivní matice vzdálenosti

- Postup:
  1. Nalezni sousední uzly *I,J* se společným předkem *K*
  2. Odstraň z matice vzdálenosti řádky a sloupce pro *I,J* a nahraď je řádkem a sloupcem pro *K*
  3. Vypočti nové vzdálenosti od uzlu *K* ke všem ostatním podle následujícího vzorce:

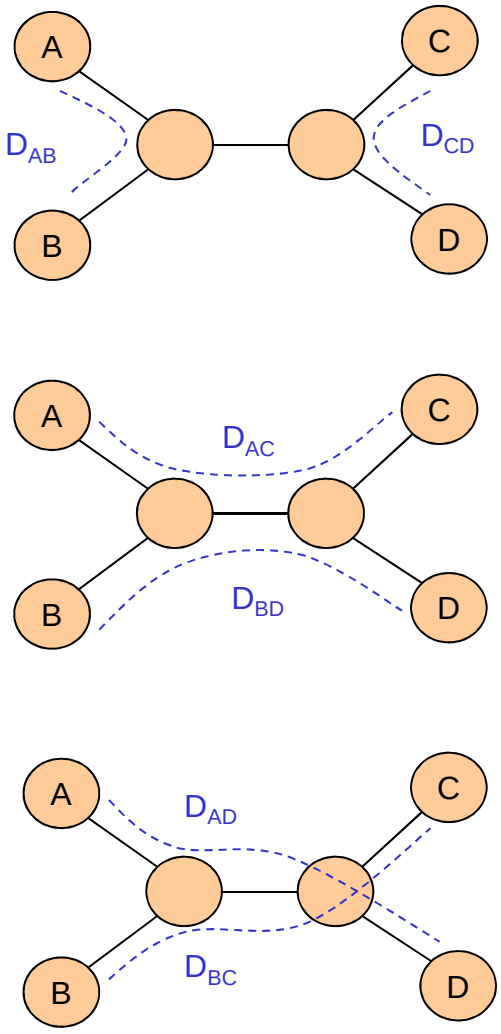
$$D_{KM} = \frac{D_{IM} + D_{JM} - D_{IJ}}{2}$$



- Problémy:
  1. Jak zjistit, že je matice aditivní?
  2. Jak vybrat sousední uzly?

Problém: Jak zjistit, že je matice aditivní?

- Čtyřbodová podmínka:
  - Založena na výpočtu tří sum:
    1. $D_{AB} + D_{CD}$
    2. $D_{AC} + D_{BD}$
    3. $D_{AD} + D_{BC}$
  - Suma 1. vyjadřuje součet délek všech hran kromě středové propojovací hrany
  - Suma 2. a 3. vyjadřuje součet délek všech hran včetně dvounásobku středové hrany



Problém: Jak zjistit, že je matice aditivní?

- Čtyřbodová podmínka je splněna, pokud:

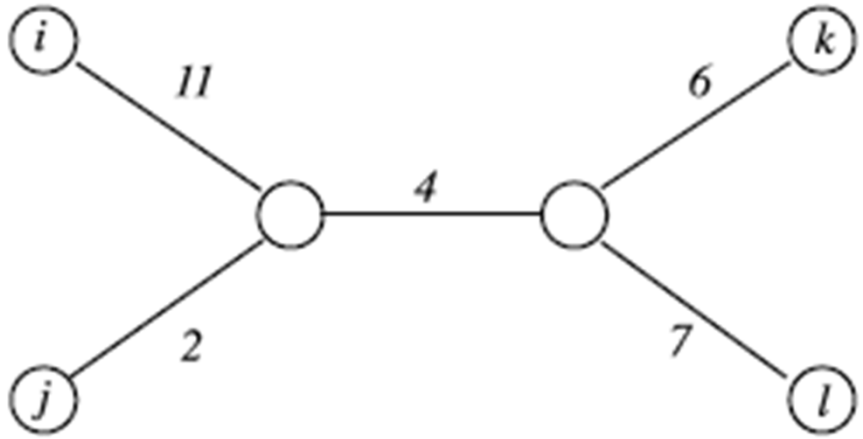
  $D_{AB}+D_{CD} \leq D_{AC} + D_{BD}$

  $D_{AB}+D_{CD} \leq D_{AD} + D_{BC}$

- Teorém:
  - *n x n* matice *D* je aditivní právě když a jen když je čtyřbodová podmínka splněna pro libovolnou čtveřici *A,B,C,D*

Problém: Jak nalézt dva sousední uzly?

- Jednoduché řešení:
  - Vybrat dvojici s nejmenší vzdáleností => špatně
  - Existují stromy, kde tato vlastnost neplatí
  - Příklad:
    - *i* a *j* jsou sousedé, ale (d_{ij} = 13) > (d_{jk} = 12)



- Obecně netriviální problém => dobré výsledky dává metoda *Neighbour-Joining*

Neighbour-Joining metoda

- N. Saitou a M.Nei (1987)
- Cílem metody je nalézt uzly *a* a *b*, které jsou:
  1. nejblíže k sobě
  2. nejvzdáleněji k ostatním
- Vzdálenost uzlů od sebe *D(a,b)* lze získat z matice vzdálenosti
- Vzdálenost uzlu od ostatních vypočtena podle vztahu:

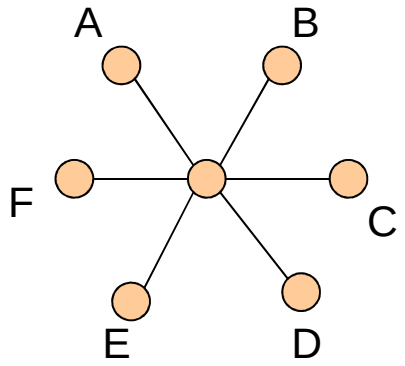
$$u(a) = \frac{1}{N-2} \sum_{i \in MnozinaUzlu} D(a,i)$$

- Hledá se dvojice s *min[D(a,b)]* a současně *max[u(a)+u(b)]*
- V některých případech nelze takovou dvojici uzlů najít jednoznačně, a proto metoda hledá minimum ze vztahu:

$$\min[D(a,b) - u(a) - u(b)]$$

Neighbour-Joining metoda

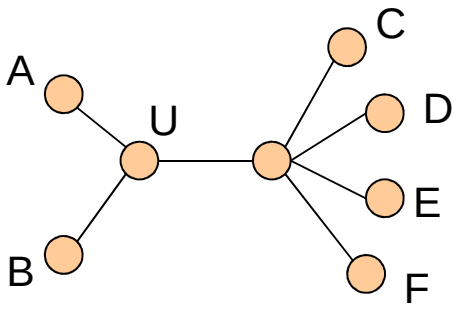
- Postup:
  1. Vypočti matici vzdáleností *D(i,j)* a vytvoř hvězdicový strom se společným středem a uzly odpovídající vstupním sekvencím



  2. Nalezni takové dva uzly *A* a *B*, které odpovídají minimu z funkce:

$$\min[D(a,b)-u(a)-u(b)]$$

a spoj uzly *A* a *B* do jednoho uzlu *U*

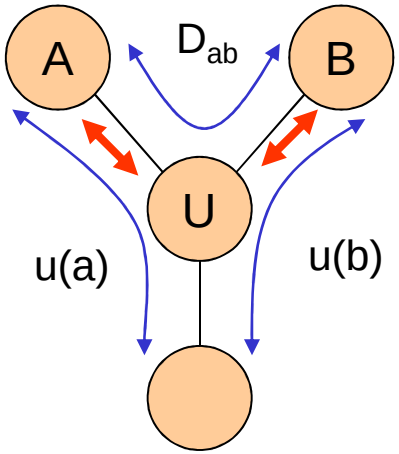


Neighbour-Joining metoda

3. Přiřaď délku nově vzniklým hranám:

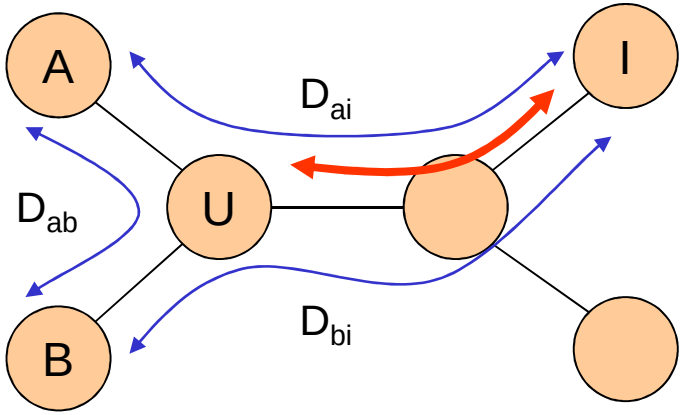
$$L(a,u) = \frac{D(a,b) + u(a) - u(b)}{2}$$

$$L(b,u) = \frac{D(a,b) + u(b) - u(a)}{2}$$



4. Vypočti vzdálenost uzlu *C* od ostatních uzlů podle vztahu:

$$D(u,i) = \frac{D(a,i) + D(b,i) - D(a,b)}{2}$$



5. Opakuj od bodu 2., dokud nezůstane jediný uzel

NJ - Příklad

- Příklad:
  - Pro zadanou matici vzdálenosti mezi řetězci *A-E* sestavte fylogenetický strom metodou NJ
- Postup:
  1. Vypočti vzdálenosti uzlu od všech ostatních:

     $u(A) = (D_{AB}+D_{AC}+D_{AD}+D_{AE}+D_{AF})/(N-2) = (5+4+7+6+8)/4 = 7.5$

     $u(B) = 10.5$

     $u(C) = 8$

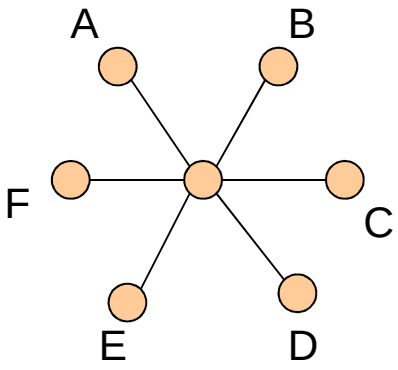
     $u(D) = 9.5$

     $u(E) = 8.5$

     $u(F) = 11$

	A	B	C	D	E
B	5				
C	4	7			
D	7	10	7		
E	6	9	6	5	
F	8	11	8	9	8

Matice vzdáleností



Počáteční strom

NJ - Příklad

2. Vytvoř matici M podle vztahu:

$M(i,j) = D_{ij} - u(i) - u(j)$

Příklad:

$M(A,B) = D_{AB} - u(A) - u(B)$
$= 5 - 7.5 - 10.5 = -13$

3. Vyber minimum z matice M:
   - Hodnota -13 mezi uzly A a B nebo D a E. => vybereme uzly A a B a sloučíme je do jednoho uzlu U

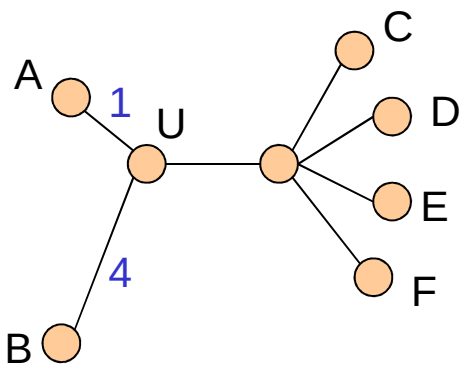
4. Vypočti délky nově vzniklých hran od uzlů A a B k uzlu U:

$L(A,U) = (D_{AB}+u(A)-u(B))/2$
$= (5+7.5-10.5)/2 = 1$

$L(B,U) = (D_{AB}+u(B)-u(A))/2$
$= (5+10.5-7.5)/2 = 4$

	A	B	C	D	E
B	-13				
C	-11.5	-11.5			
D	-10	-10	-10.5		
E	-10	-10	-10.5	-13	
F	-10.5	-10.5	-11	-11.5	-11.5

Matice M (minimalizační funkce)



Upravený strom

NJ - Příklad

5. Vypočti vzdálenost uzlu C od všech ostatních a sestav novou matici vzdálenosti

$D'_{CU} = (D_{AC}+D_{BC}-D_{AB})/2 = 3$

$D'_{DU} = (D_{AD}+D_{BD}-D_{AB})/2 = 6$

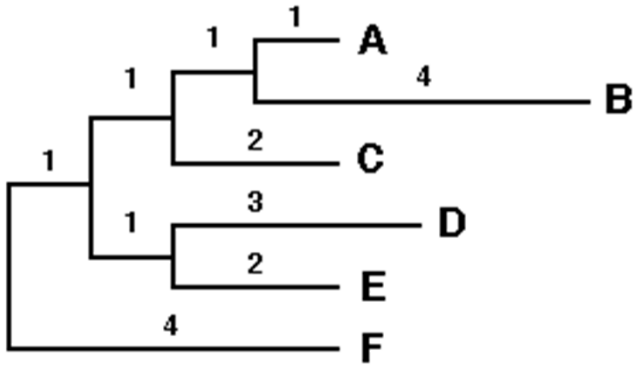
$D'_{EU} = (D_{AE}+D_{BE}-D_{AB})/2 = 5$

$D'_{FU} = (D_{AF}+D_{BF}-D_{AB})/2 = 7$

	U	C	D	E
C	3			
D	6	7		
E	5	6	5	
F	7	8	9	8

6. Pokračuj ve výpočtu dokud neobdržíš celý strom

- Výsledný strom:



NJ – určení kořene

- Metoda NJ generuje obecně nekořenové stromy
- Pro určení kořene se do skupiny sekvencí vloží tzv. *outgroup*, nebo-li jedinec, který je velmi vzdálený od skupiny ostatních analyzovaných sekvencí (ingroups)
  - Např. ryba pro skupinu savců
  - Outgroup bude připojen ke všem ostatním nejdelší hranou, na kterou se vloží kořen stromu.
- Pokud *outgroup* jedince nelze jednoduše vybrat, potom se vybere nejdelší hrana stromu a kořen se vloží doprostřed této hrany.

Unweighted **P**air **G**roup **M**ethod with **A**rithmetic mean

- Postup:
  - Vyber z tabulky podobnosti sekvenci s_A a s_B s nejvyšší mírou identity
  - Dvojici spoj do jedné sekvence s_{AB} a přepočítej vzdálenosti v tabulce podobnosti podle vztahu: $D_{AB,C} = (D_{A,C}+D_{B,C})/2$
  - Spojení obou sekvencí zakresli do pomocného stromu
  - Opakuj, dokud nejsou spojeny všechny dvojice

- Příklad:
  - Mějme následující tabulku vzdáleností (shodná s předchozím příkladem)

	A	B	C	D	E
B	5				
C	4	7			
D	7	10	7		
E	6	9	6	5	
F	8	11	8	9	8

  - Sloučíme sekvence A a C a přepočítáme tabulku. Už v prvním kroku výpočtu došlo ke špatnému sloučení uzlů A a C!

UPGMA (pokračování)

- Přepočet vzdáleností
  - $D_{AC,B} = (D_{A,B} + D_{B,C})/2 = (5+7)/2 = 6$
  - $D_{AC,D} = (D_{A,D} + D_{C,D})/2 = (7+7)/2 = 7$
  - $D_{AC,E} = (D_{A,E} + D_{C,E})/2 = (6+6)/2 = 6$
  - $D_{AC,F} = (D_{A,F} + D_{C,F})/2 = (8+8)/2 = 8$
- Upravená tabulka:

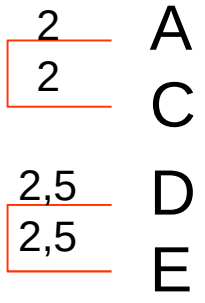
	U_{AC}	B	D	E
B	6			
D	7	10		
E	6	9	5	
F	8	11	9	8

- Sloučíme D a E

- Původní tabulka

	A	B	C	D	E
B	5				
C	4	7			
D	7	10	7		
E	6	9	6	5	
F	8	11	8	9	8

- Strom



- Výpočet délek
  - $L_{A,C} = 4/2 = 2$
  - $L_{D,E} = 5/2 = 2,5$

UPGMA (pokračování)

- Přepočet vzdáleností
  - $D_{DE,AC} = (D_{D,AC} + D_{E,AC})/2 = (7+6)/2 = 6,5$
  - $D_{DE,B} = (D_{D,B} + D_{E,B})/2 = (10+9)/2 = 9,5$
  - $D_{DE,F} = (D_{D,F} + D_{E,F})/2 = (9+8)/2 = 8,5$
- Upravená tabulka:

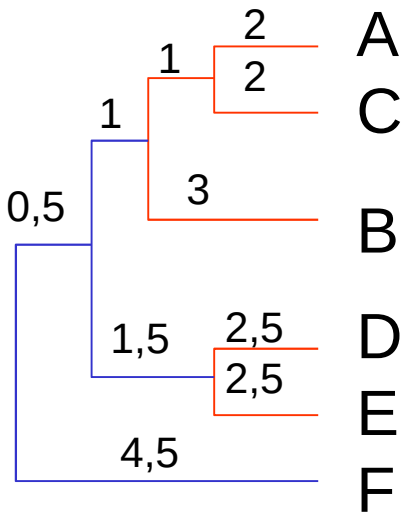
	U_{AC}	B	U_{DE}
B	6		
U_{DE}	6,5	9,5	
F	8	11	8,5

- Sloučíme U_{AC} a B
- Výpočet délek
  - $L_{AC,B} = 6/2 = 3$

- Původní tabulka

	U_{AC}	B	D	E
B	6			
D	7	10		
E	6	9	5	
F	8	11	9	8

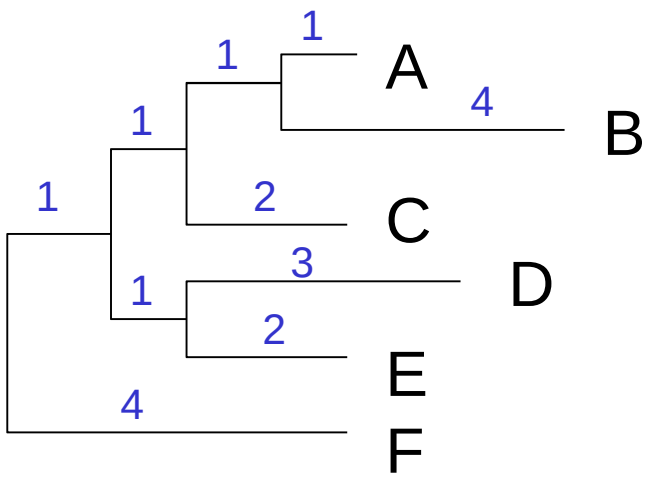
- Výsledný strom



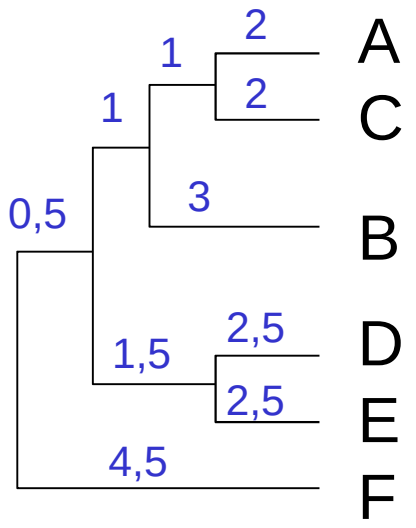
NJ vs. UPGMA

- Nevýhody UPGMA:
  - Algoritmus produkuje *ultrametrické stromy*: vzdálenost od kořene ke všem uzlům je stejná
  - UPGMA předpokládá *konstantní molekulární hodiny* ve všech větvích stromu

Neighbour-Joining

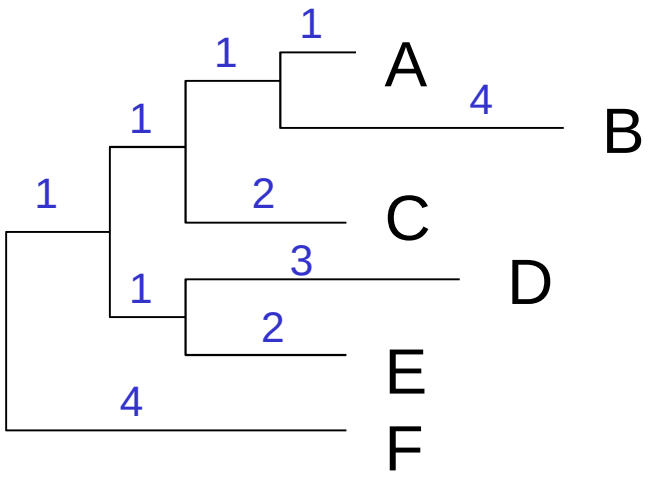


UPGMA

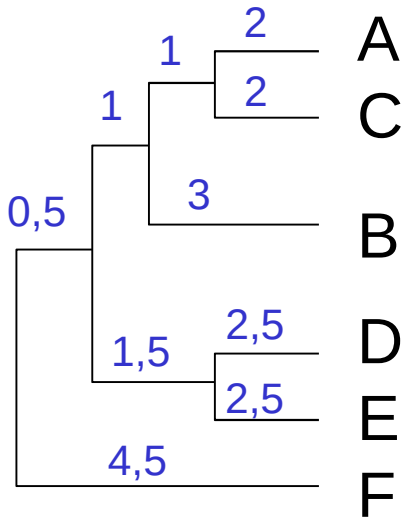


NJ vs. UPGMA

- Který strom je více odpovídá původnímu záměru?
  - jednoduchá kontrola vypočtením matice vzdálenosti



	A	B	C	D	E
B	5				
C	4	7			
D	7	10	7		
E	6	9	6	5	
F	8	11	8	9	8



	A	B	C	D	E
B	6				
C	4	6			
D	8	8	8		
E	8	8	8	5	
F	9	9	9	9	9

Pro neaditivní matice vzdálenosti

- Pokud je matice neaditivní, hledáme takový strom *T*, který nejlépe *aproximuje* hodnoty *D* v matici vzdálenosti
- Cílem je minimalizovat chybu (čtverce):

$$square\_error = \sum_{i,j} (d_{ij}(T) - D_{ij})^2$$

- Problém nalezení nejmenšího čtverce chyby při konstrukci fylogenetického strom je obecně NP-úplný
- Avšak metoda *Neighbour-Joining* dosahuje velmi dobrých výsledků pro neaditivní matice

Shrnutí: Metody založené na vzdálenosti

- Konstrukce stromu vychází z matice vzdálenosti *M*
- Cílem metod je takový strom, kde stromová vzdálenost odpovídá vzdálenosti z matice *M* tj. $D_{ij} = d_{ij}(T)$
- Strom lze takovýto strom sestrojit, tj. je splněna *4-bodová podmínka*, označuje se jako *aditivní* a lze sestrojit metodou *UPGMA* nebo lépe *NJ*
- Pro *neaditivní stromy* je možné najít pouze přibližné řešení. Nejlepší aproximaci získáme metodou *nejmenších čtverců*. Tento problém je však *NP-těžký*, a proto se používá spíše *NJ*, která dosahuje velmi dobrých výsledků

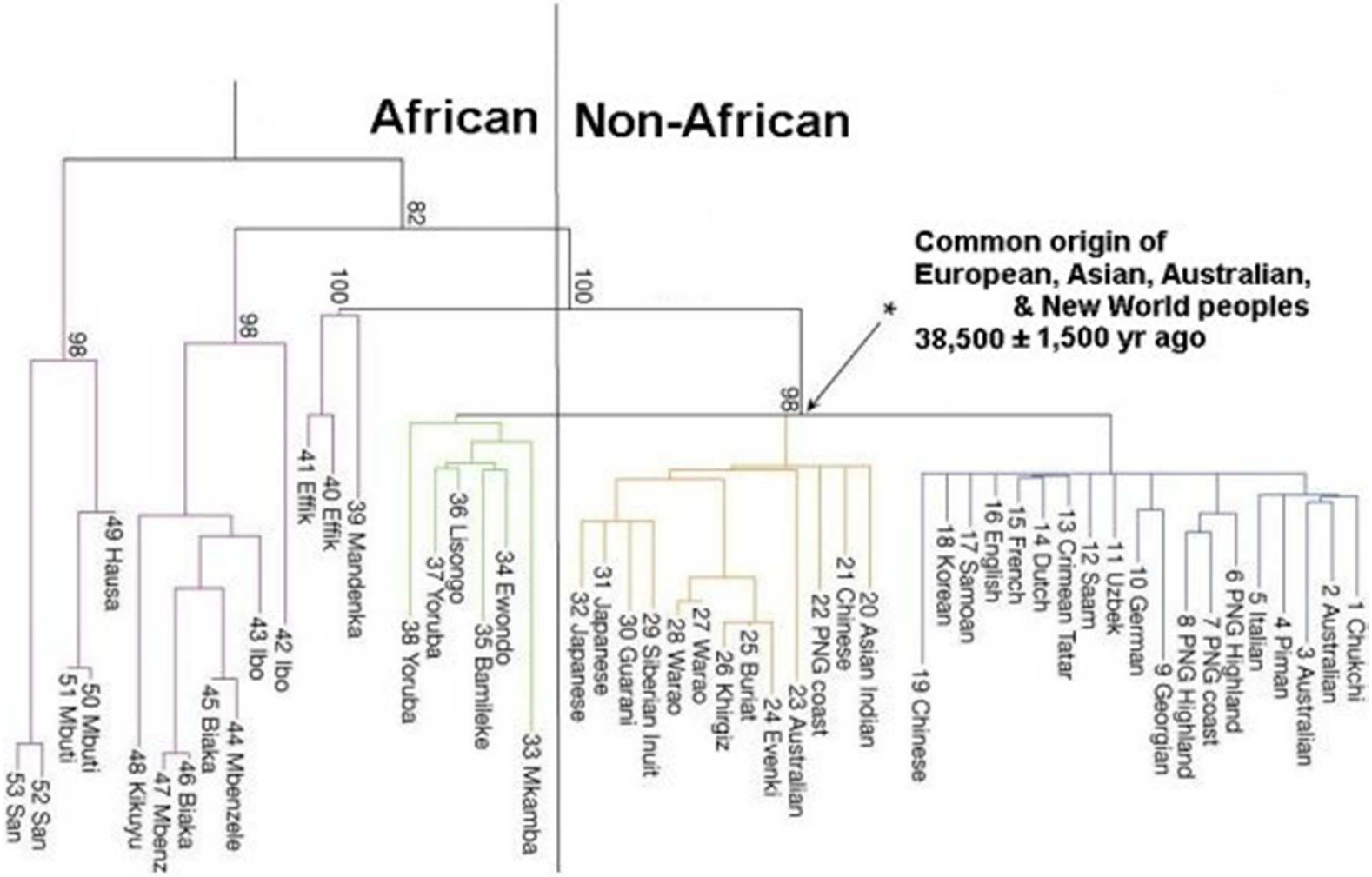
Hádanka o původu lidstva

- „Out of Africa“ hypotéza
  - Vývoj lidí začal v Africe zhruba před 150,000 lety
  - Lidé migrovali z Afriky a vytvářeli ostatní rasy po celém světě
  - Není žádná přímá souvislost s neandrtálci
- Multi-regionální hypotéze
  - Lidstvo se vyvíjelo v posledních 2. mil. letech jako jeden druh nezávisle v různých oblastech
  - Lidé pocházející z Afriky se postupně promíchali s ostatními rasami
  - Existuje souvislost mezi člověkem a neandrtálcem

Hádanka o původu lidstva

- Analýza mtDNA se přiklání spíše „Out of Afrika“ hypotéza
- Důvody:
  - Africká populace obsahuje nejvíce změn (pod-populace měli více času pro vytvoření těchto změn)
  - Evoluční strom oddělil Afričany od všech zbývajících populací
  - Kořen stromu byl umístěn na větvi mezi skupinami s nejvyšším výskytem změn

Evoluční strom vývoje člověka



Migrace člověka z Afriky

1. Yorubans
2. Western Pygmies
3. Eastern Pygmies
4. Hadza
5. !Kung

Spojitost s neandrtálci?

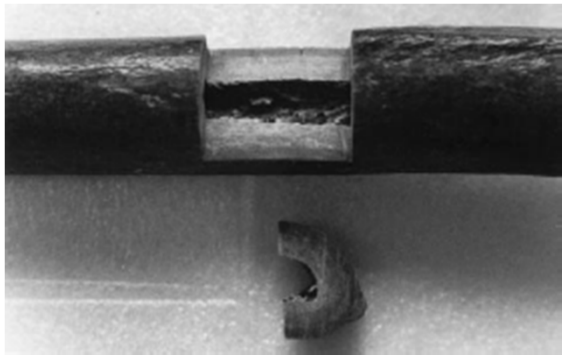
- Objevení dvou neandrtálců
  1. Feldhofer, Německo
  2. Mezmaiskaya, Kavkaz
  – Vzdálenost 2500km

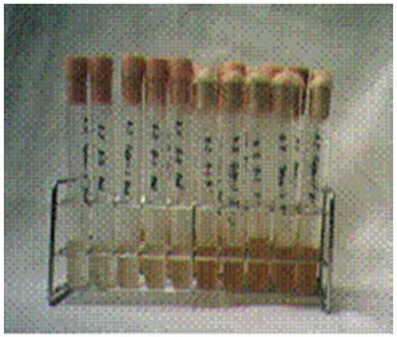
Spojitost s neandrtálci?

- Existuje spojitost mezi neandrtálci a současnými Evropany?

Figure 1. Illustration of Neandertal Man
(Reprinted by permission from John Gurche/National Geographic.)

Sekvenování neandrtálců





- Bylo použito mtDNA z kostí, protože se vyskytuje v cca 1000x větším množství než jaderná DNA
- DNA se v průběhu času rozkládá a jen malá část může být obnovena (limit cca 100,000 let)
- Provedeno rozmnožení mtDNA použitím PCR

Neandrtálec vs. člověk

- Byly zjištěna překvapivě velké rozdíly
- Neandrtálec vs. člověk
  - 22 substitucí a 6 insdelů v oblasti 357 bp
- Člověk vs. člověk
  - pouze 8 substitucí
- Pokud nepocházíme z neandrtálců, potom kdo je náš předchůdce?

Figure 1. Illustration of Neandertal Man
(Reprinted by permission from John Gurche/National Geographic.)

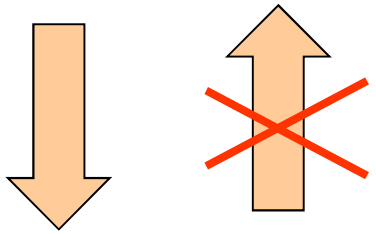
×

Vícenásobné zarovnání vs. Matice vzdálenosti

1. Z vícenásobného zarovnání lze sestrojit matice vzdáleností
2. Naopak je to ale problém

- Při kroku 1. se ztrácí určitá informace, vzniká myšlenka: Proč nevytvářet fylogenetické stromy přímo na základě vícenásobného zarovnání?
- Vznikají metody založené na znacích (Parsinomy)

A: **AC-GCGG-C**
B: **AC-GC-GAG**
C: **GCCGC-GAG**
...



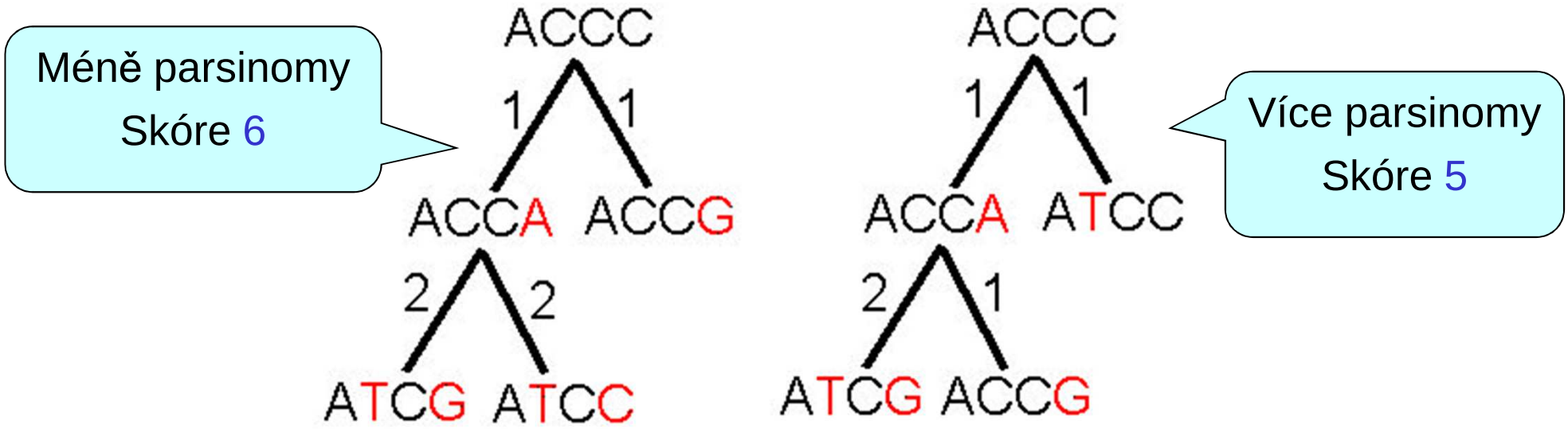
	A	B	C	D
B	3			
C	6	7		
D	5	6	5	
E	7	8	9	8

Metody založené na znacích

- Vstup:
  - Vícenásobné zarovnání sekvencí zkoumaných organizmů
- Výstup:
  - Fylogenetický strom s takovými vnitřními uzly, které nejlépe odpovídají znakům sekvencí organizmů na listových uzlech
- Pokud ohodnotíme hrany Hamingovou vzdáleností mezi uzly, suma ohodnocených hran nám udává celkové tzv. parsinomy skóre pro sestavený strom
- Cílem je nalézt takový strom, který má minimální parsinomy skóre

Metody založené na znacích

- Příklad: sekvence ATCG, ATCC, ACCG



- Problémy:
  1. Jak nalézt tvar stromu?
  2. Jak nalézt ohodnocení vnitřních uzlů?

Malý parsinomy problém

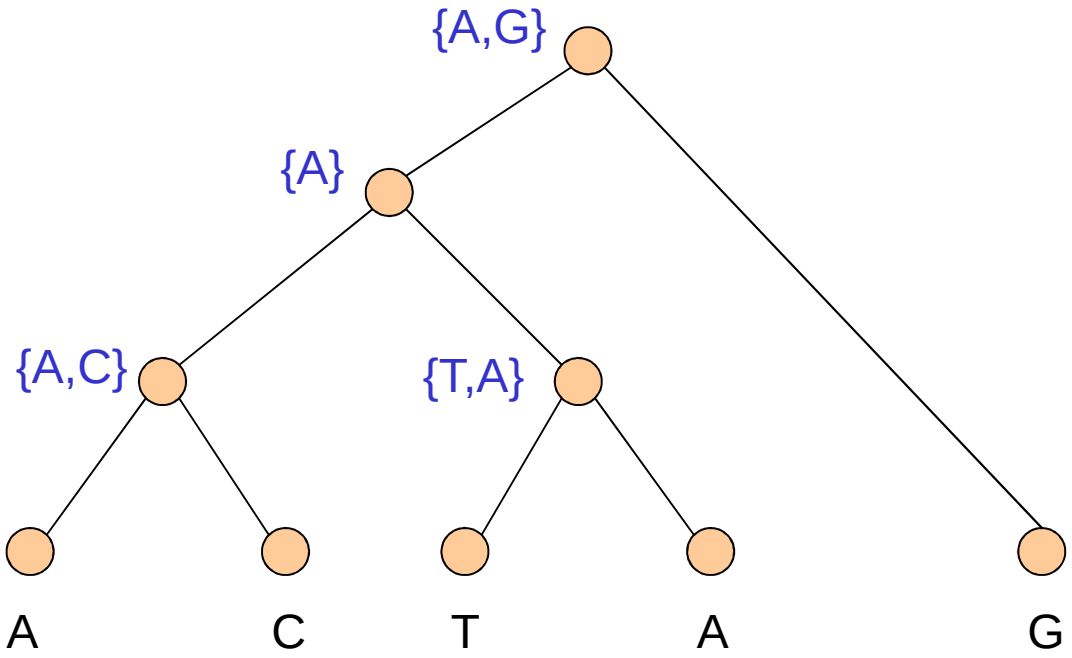
- Vstup:
  - Vytvořený strom, kde listové uzly tvoří sekvence organizmů
- Výstup:
  - Označení vnitřních uzlů tak, aby parsinomy skóre celého stromu bylo minimální
- Poznámka:
  - Jednotlivé znaky vnitřních uzlů lze v tomto případě posuzovat odděleně, protože jsou na sobě nezávislé

Fitchův algoritmus

- Řeší malý parsinomy problém – nalezne optimální přirazení znaků vnitřním uzlům
- Využívá myšlenky dynamického programování: rekurzivní přístup, kdy skóre rodiče je vypočteno na základě skóre obou potomků
- Krok 1: výpočet od listových uzlů ke kořenu
  - Pro každý vnitřní uzel vypočti množinu přípustných znaků, podle následujícího pravidla:
    - pokud množiny potomků se překrývají, potom vyber jejich průnik
    - jinak vyber jejich sjednocení

Fitchův algoritmus

- Příklad:

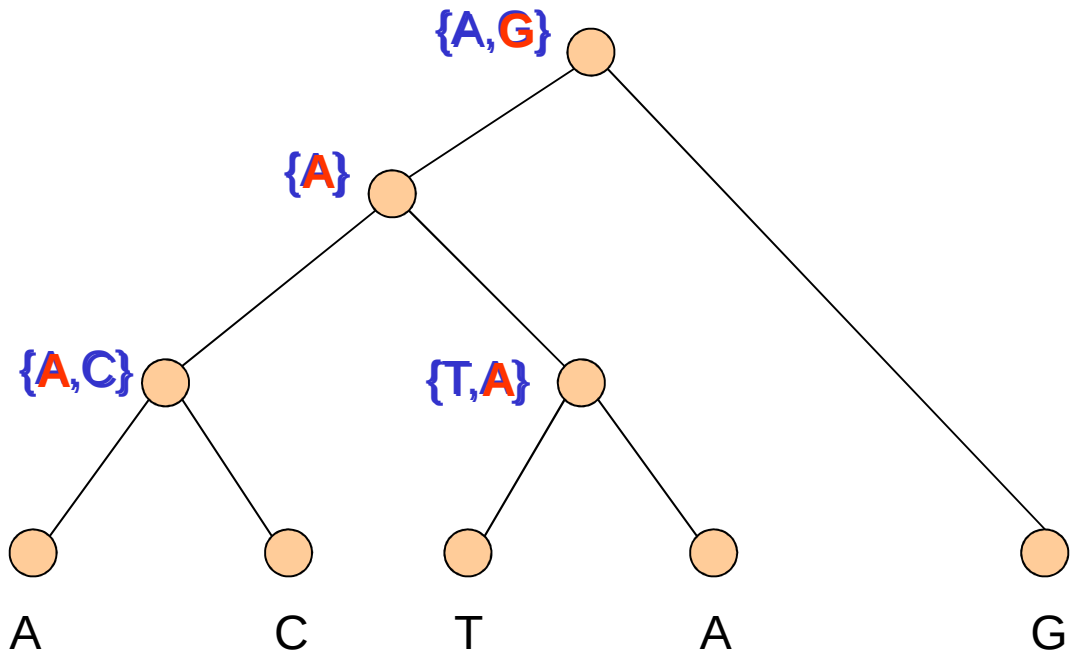


Fitchův algoritmus

- Krok 2: Zpětný chod od kořene k listovým uzlům
  - Pro kořen vyber libovolný znak z množiny
  - Pro vnitřní uzly aplikuj následující pravidlo:
    - pokud množina obsahuje stejný znak, jako jeho předek, potom vyber tento znak také
    - jinak vyber libovolný znak

Fitchův algoritmus

- Pokračování příkladu:



Neváhovaný vs. Váhovaný

- Doposud jsme používali Hammingovu vzdálenost pro ohodnocení stromu:

  $d_H(v, w) = 0$ if $v=w$

  $d_H(v, w) = 1$ otherwise

- Pro dosažení dobrých reálních výsledků je to však nepřesné schéma – změny mezi jednotlivými znaky A, C, G, T mají různou pravděpodobnost
- Je potřeba použít obecnou skórovací matici pro výpočet vzdálenosti vnitřních uzlů
- Zobecnění malého parsinomy problému na malý váhovaný parsinomy problém

Neváhovaný vs. Váhovaný

- Skórovací matice:

Neváhovaný parsimony problém

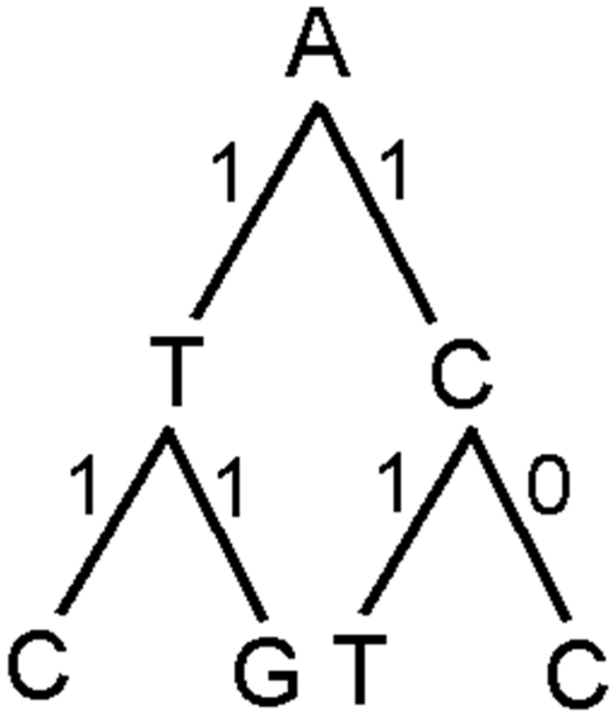
	A	T	G	C
A	0	1	1	1
T	1	0	1	1
G	1	1	0	1
C	1	1	1	0

Váhovaný parsimony problém

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

Neváhovaný vs. Váhovaný

- Neváhovaný parsinomy problém



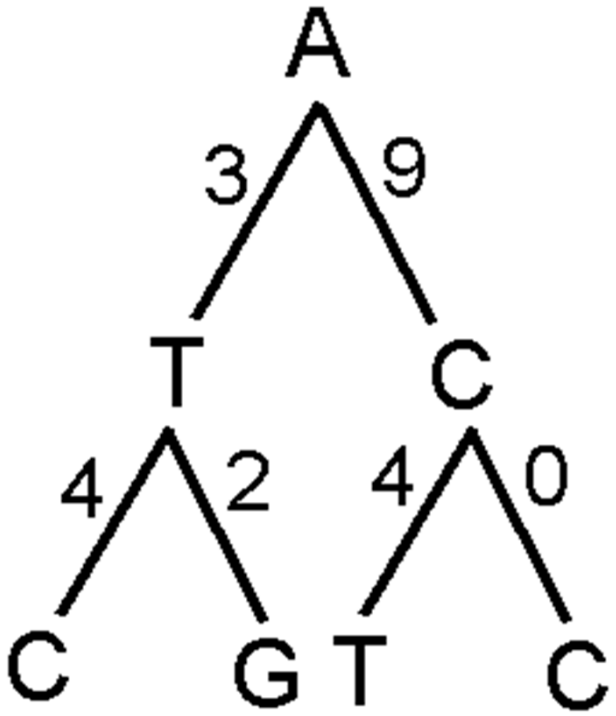
Skórovací matice

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

Parsinomy skóre: 5

Neváhovaný vs. Váhovaný

- Váhovaný parsinomy problém



Skórovací matice

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

Parsinomy skóre: 22

Malý váhovaný parsinomy problém

- Vstup:
  - Vytvořený strom, kde každý listový uzel je ohodnocen znakem vstupní abecedy o velikosti *k*-znaků
  - Skórovací matice δ_{ij} o velikosti *k x k* položek
- Výstup:
  - Ohodnocení vnitřních uzlů stromu tak, aby bylo dosaženo minimální váhované parsinomy skóre

Sankoffův algoritmus

- Krok 1: Výpočet od listových uzlů ke kořenu
  1. Ke každému uzlu vytvoř tabulku skóre pro všechny možné znaky
  2. Jdi od listových uzlů ke kořeni a doplňuj tabulky skóre podle následujících pravidel:

     Listový uzel:

     - jestliže se znak shoduje, potom skóre = 0 jinak ∞ (nekonečno)

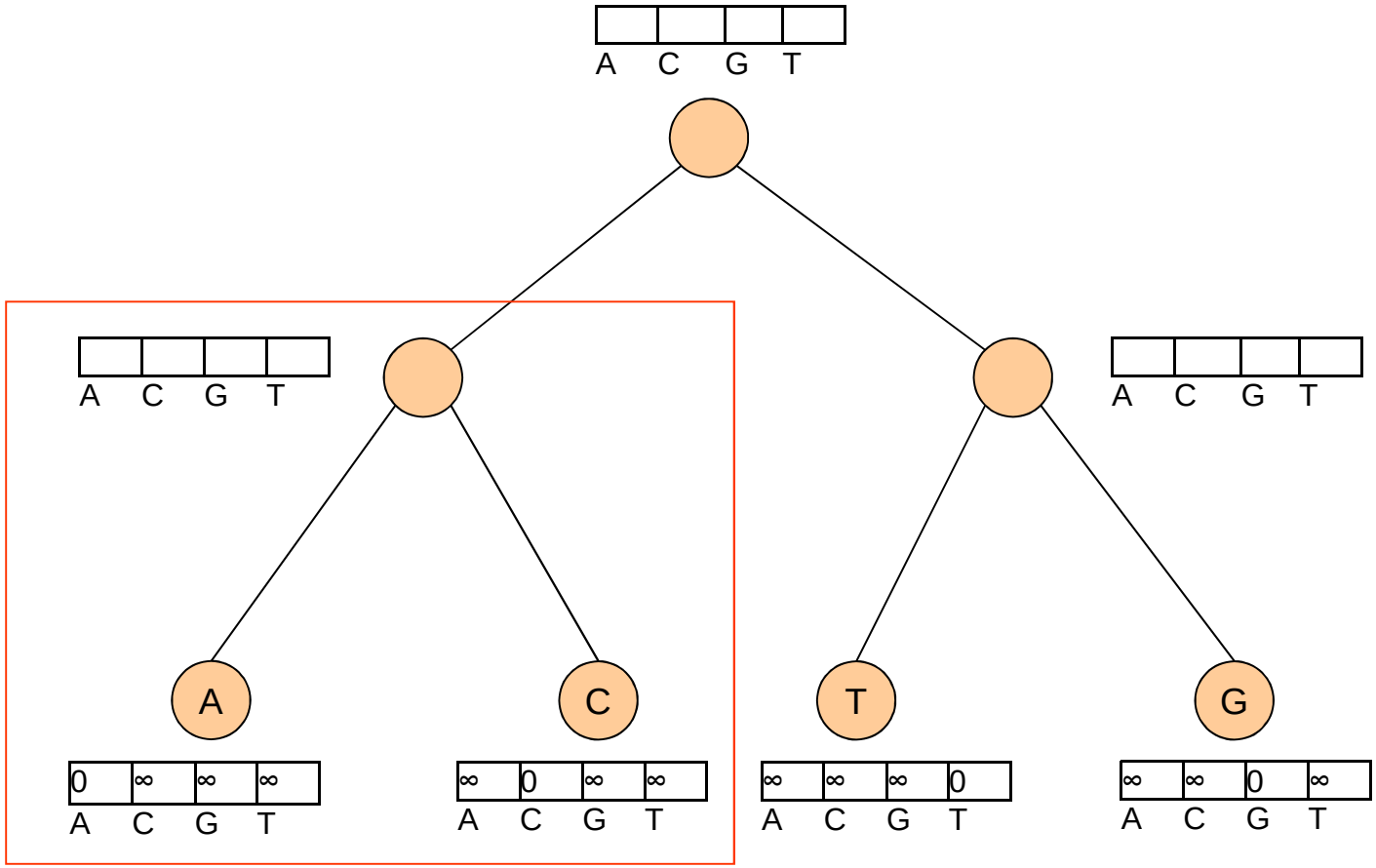
     Nelistový uzel:

     - skóre je vypočteno jako

$$s_t(\textbf{rodič}) = \min_i \{s_i(\textbf{levý potomek}) + \delta_{i\,t}\} + \min_j \{s_j(\textbf{pravý potomek}) + \delta_{j,t}\}$$

Sankoffův algoritmus

- Příklad:



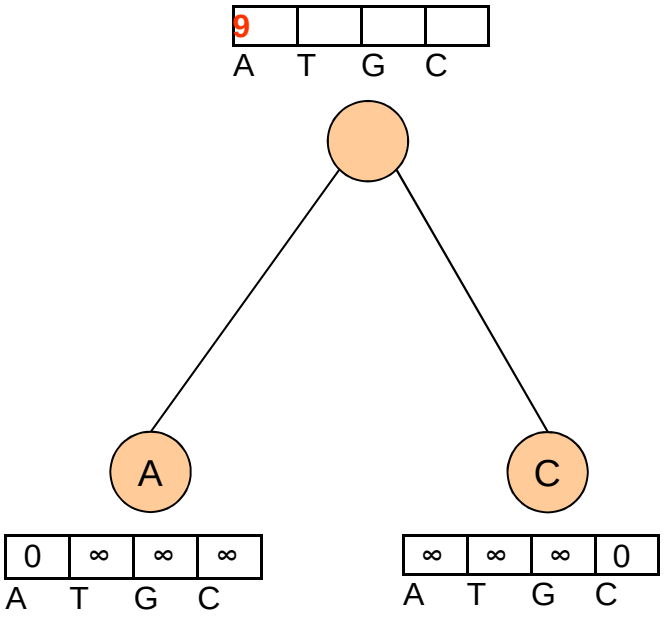
Sankoffův algoritmus

- Příklad:

$$s_t(v) = \min_i \{s_i(u) + \delta_{i,t}\} + \min_j \{s_j(w) + \delta_{j,t}\}$$
$$= 0 + 9 = 9$$

Skórovací matice

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



Levý podstrom

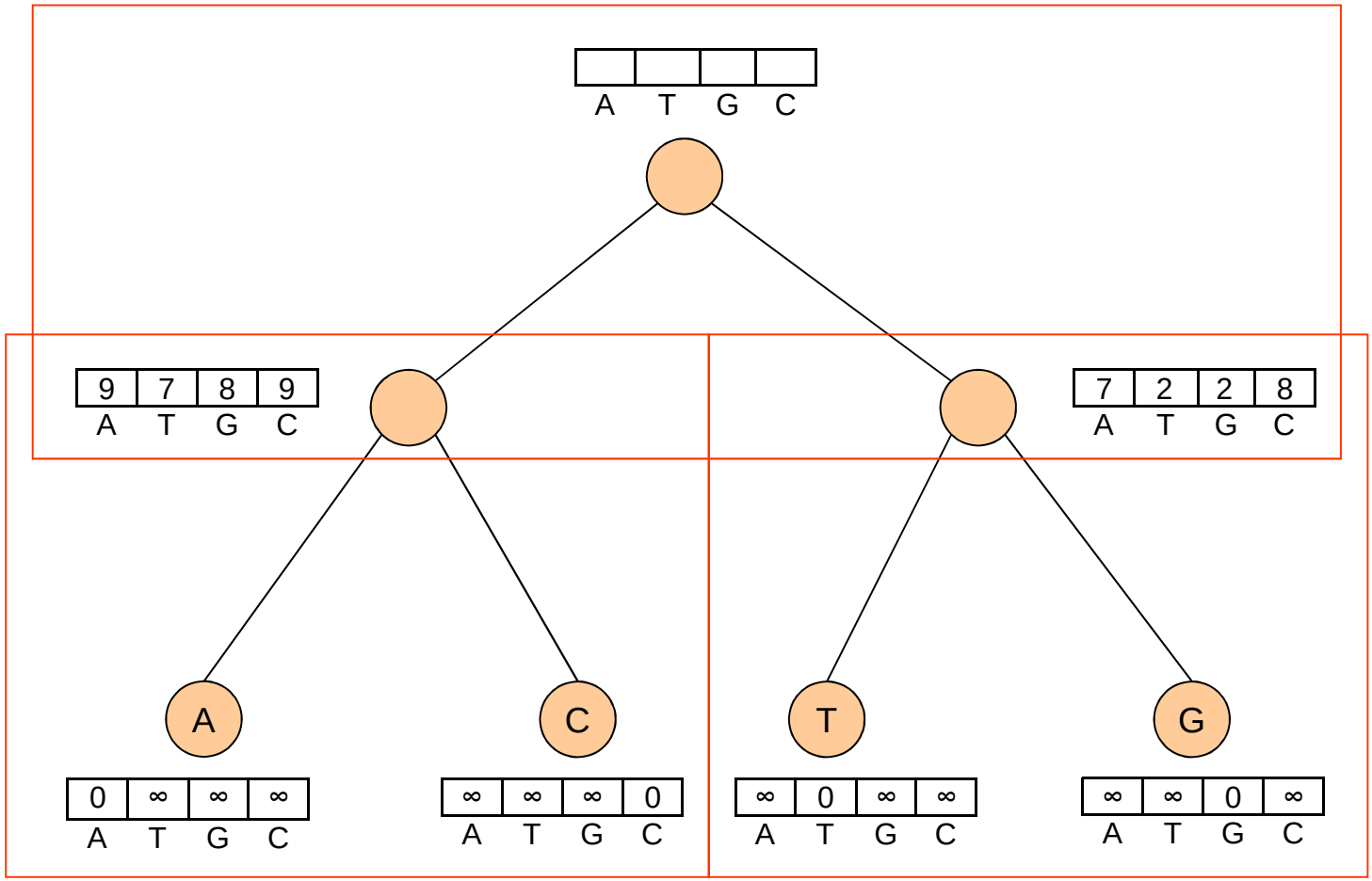
	$s_i(u)$	$\delta_{i,A}$	sum
A	0	0	**0**
T	∞	3	∞
G	∞	4	∞
C	∞	9	∞

Pravý podstrom

	$s_j(u)$	$\delta_{j,A}$	sum
A	∞	0	∞
T	∞	3	∞
G	∞	4	∞
C	0	9	**9**

Sankoffův algoritmus

- Příklad:



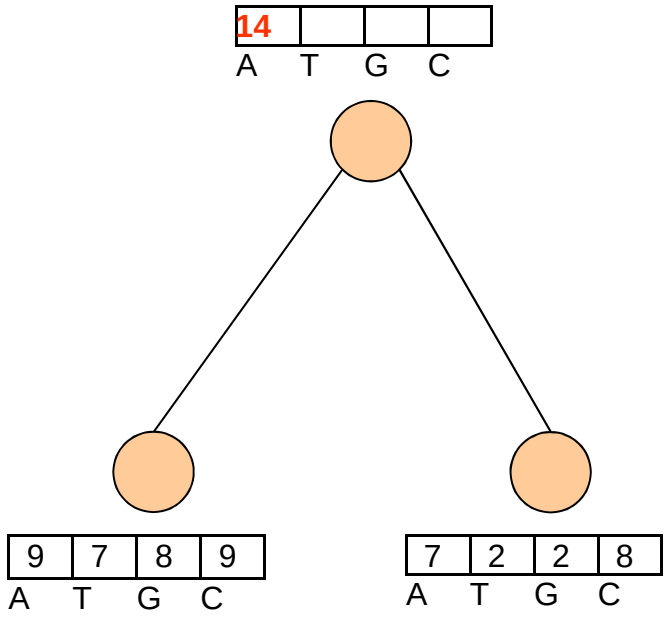
Sankoffův algoritmus

- Příklad:

$$s_t(v) = \min_i \{s_i(u) + \delta_{i,t}\} + \min_j \{s_j(w) + \delta_{j,t}\}$$
$$= 9 + 5 = 14$$

Skórovací matice

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



Levý podstrom

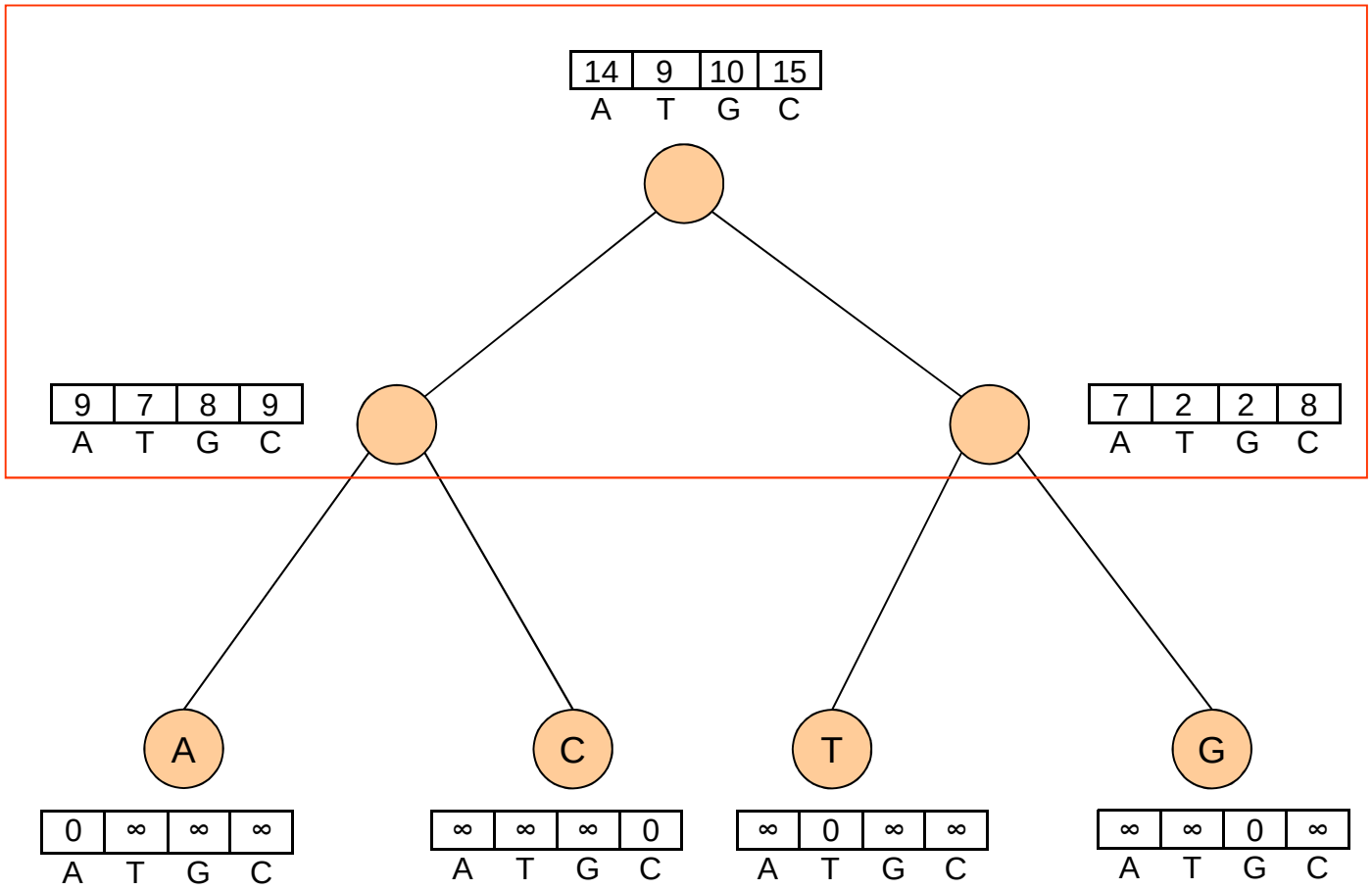
	$s_i(u)$	$\delta_{i,A}$	sum
A	9	0	**9**
T	7	3	10
G	8	4	12
C	9	9	18

Pravý podstrom

	$s_j(u)$	$\delta_{j,A}$	sum
A	7	0	7
T	2	3	**5**
G	2	4	6
C	8	9	17

Sankoffův algoritmus

- Příklad:



Sankoffův algoritmus

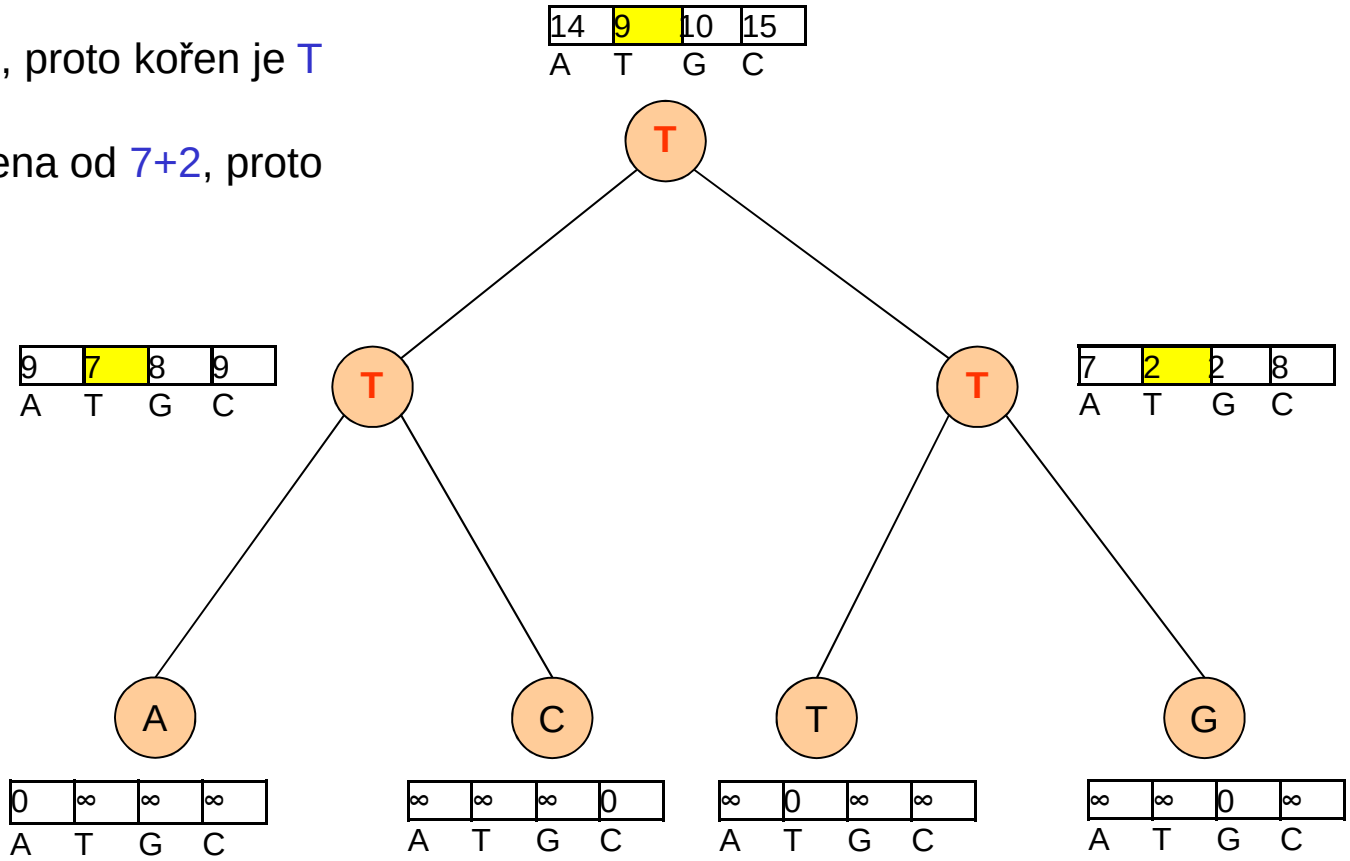
- Krok 2: Zpětný chod od kořene k listům
  1. Nejmenší skóre v kořenu stromu odpovídá nejmenšímu parsinomy skóre celého stromu
  2. Pokračuj od kořene k listům a pro každý vnitřní uzel vyber takový znak, který odpovídá skóre získanému v rodičovském uzlu.

Sankoffův algoritmus

- Pokračování příkladu:

Nejmenší skóre je 9, proto kořen je T

Hodnota 9 je odvozena od 7+2, proto
levý potomek je T,
pravý potomek je T



Fitch vs. Sankoff

- Oba mají časovou složitost $O(kn)$
- Lze dokázat, že pokud pro Sankoffův algoritmus použijeme neváhovanou skórovací matici, potom dává stejné výsledky jako Fitchův algoritmus
- Sankoffův algoritmus je zobecněnou verzí Fitchova algoritmu

Velký parsinomy problém

- Vstup:
  - Matice *n x m* vícenásobného zarovnání popisující *n* organizmů reprezentovaných jako řetězec o velikosti *m* znaků
- Výstup:
  - Strom *T* s n listovými uzly odpovídající řádkům matice vícenásobného zarovnání a vnitřními uzly označenými tak, aby parsinomy skóre stromu bylo minimální přes všechny možné stromy a všechny možné ohodnocení jejich vnitřních uzlů

Velký parsinomy problém

- Stavový prostor problému je obrovský
  - možných kořenových stromů: $\frac{(2n-3)!}{2^{n-2}(n-2)!}$
  - možných nekořenových stromů: $\frac{(2n-5)!}{2^{n-3}(n-3)!}$
- Jedná se o NP úplný problém
  - prohledat celý stavový prostor je možné pouze pro n<10
- Musíme použít heuristiku

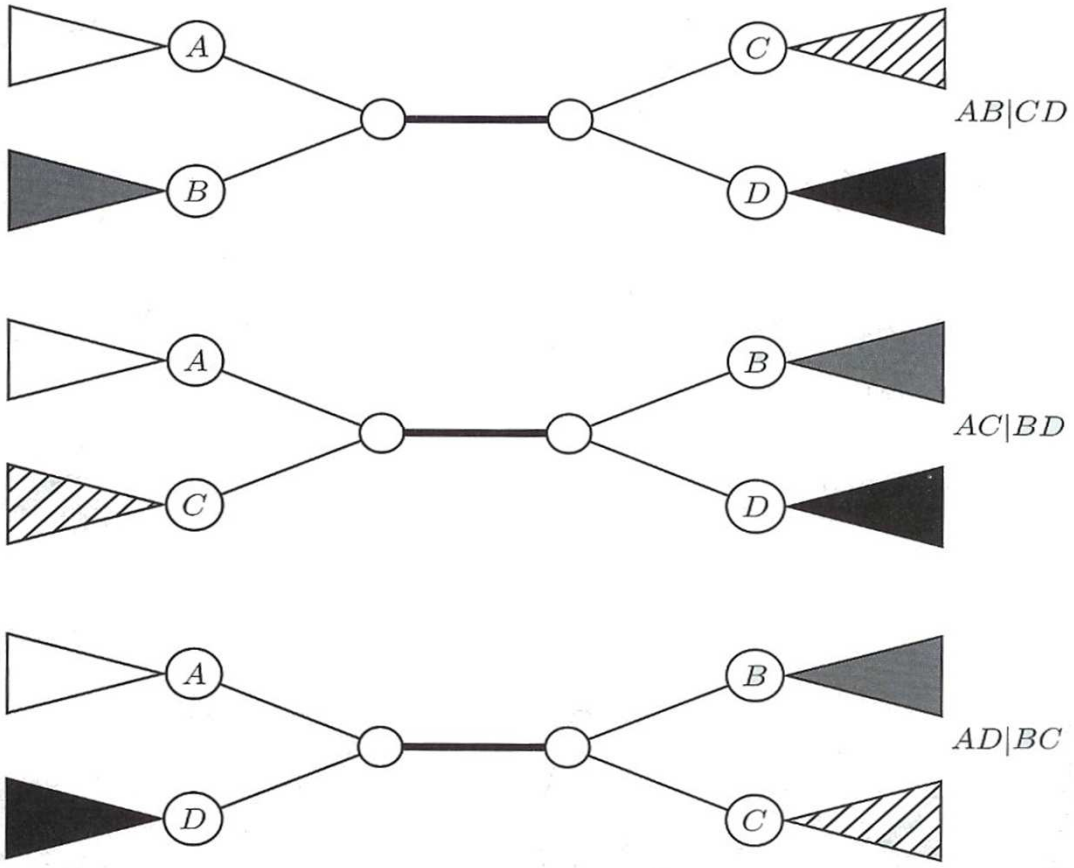
Listů	Nekořenových stromů	Kořenových stromů
3	1	3
4	3	15
5	15	105
6	105	945
7	945	10395
8	10395	135135
9	135135	2027025
10	2027025	2.22E+020
20	2.22E+020	8.20E+021

Záměna nejbližších sousedů

- Původní název: Nearest neighbor interchange
- Pro každou hranu stromu lze sestrojit obecné schéma nekořenového stromu obsahujícího 4 podstromy ABCD
- Tyto 4 podstromy mohou být kombinovány třemi možnými způsoby
- Tyto tři stromy jsou označovány jako sousedé, které je možné mezi sebou transformovat z jednoho na druhý

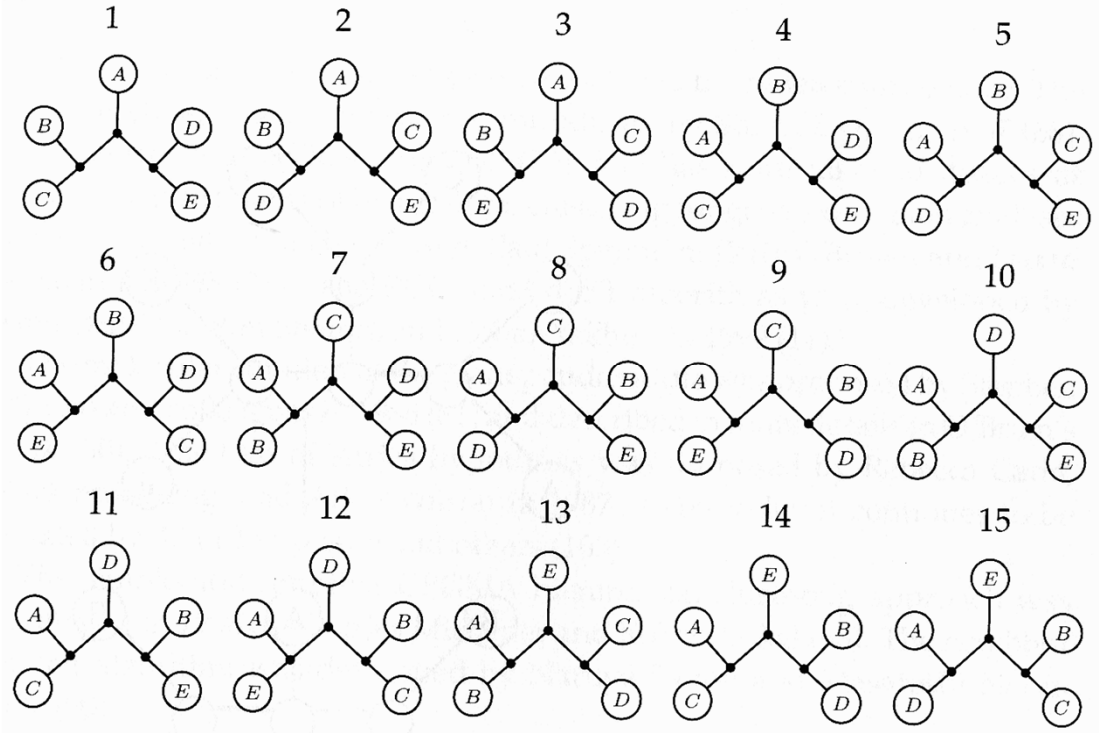
Záměna nejbližších sousedů

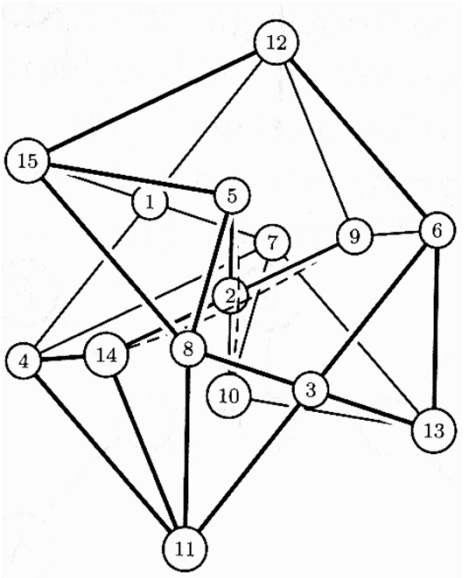
- Obecné schéma sousedních stromů



Záměna nejbližších sousedů

- Pokud bychom si vzali všechny možné stromu, potom lze sestrojit graf, kde vrcholy označují strom a hrany propojují takové stromy, které jsou sousedy
- Příklad stromu s 5 uzly:



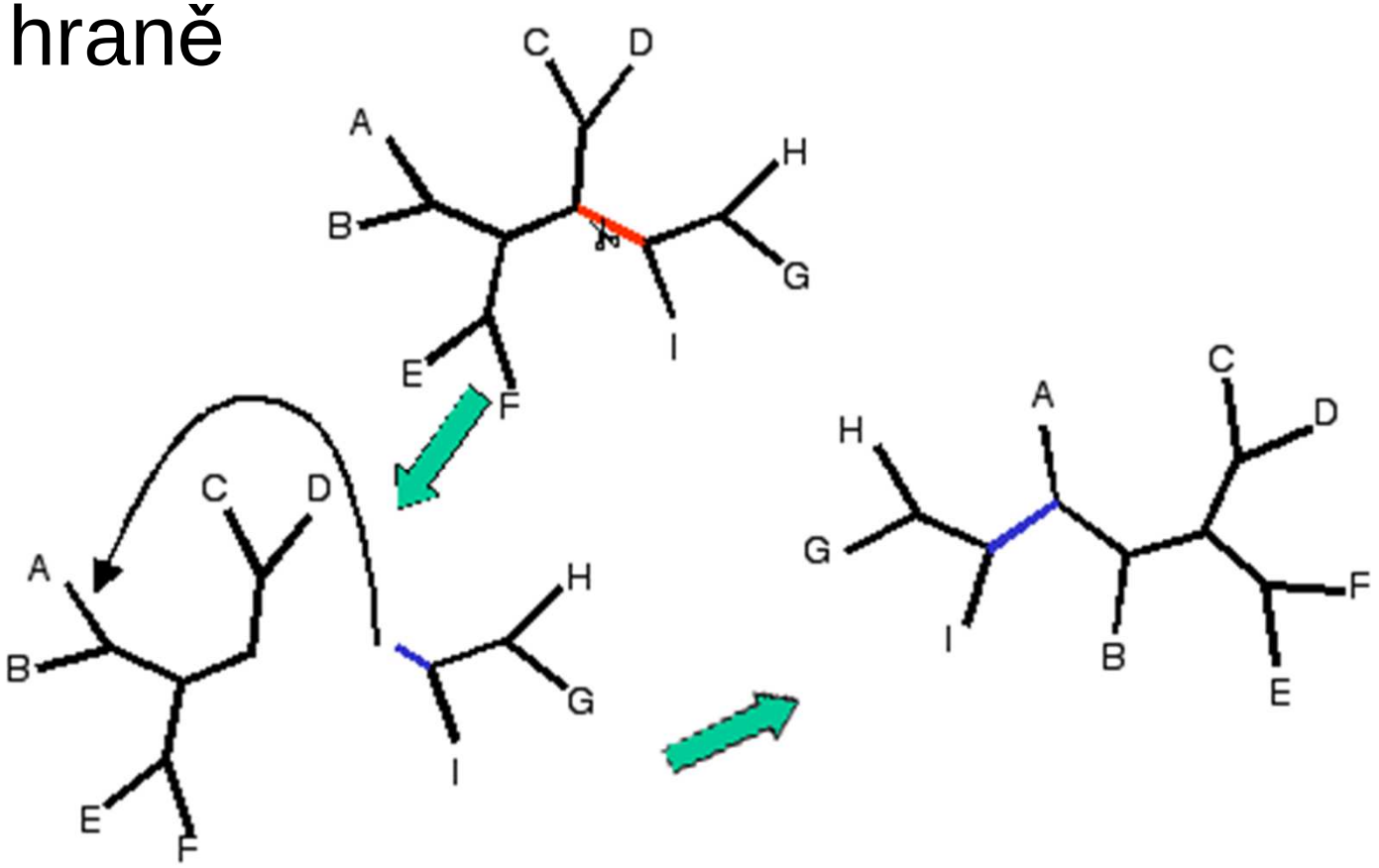


Základní heuristika

- Postup:
  1. Začni v libovolném uzlu grafu sousedů
  2. Pro všechny své sousedy vypočítej parsinomy skóre
  3. Přemísti se do takového uzlu, který má nejmenší parsinomy skóre
- Poznámky:
  - Nelze zaručit nalezení nejlepšího stromu
  - Obvykle algoritmus skončí v lokálním minimu
  - Je potřeba použít další techniky

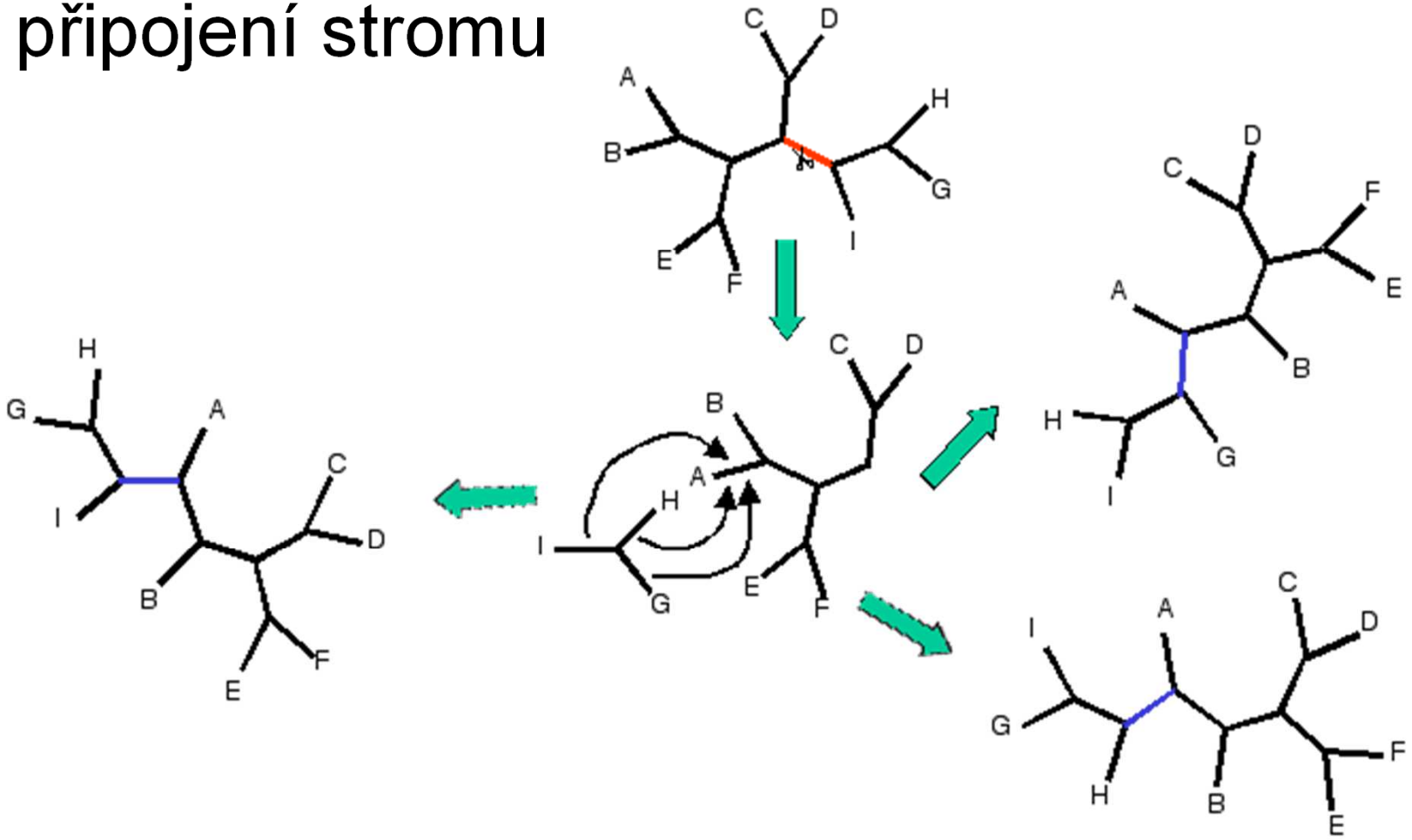
Rozpůlení stromu a jeho reorganizace I.

- Strom se rozpůlí na vybrané hraně a připojí na jiné místo



Rozpůlení stromu a jeho reorganizace II.

- Zkouší se více možností pro připojení stromu



Shrnutí: Metody založené na znacích

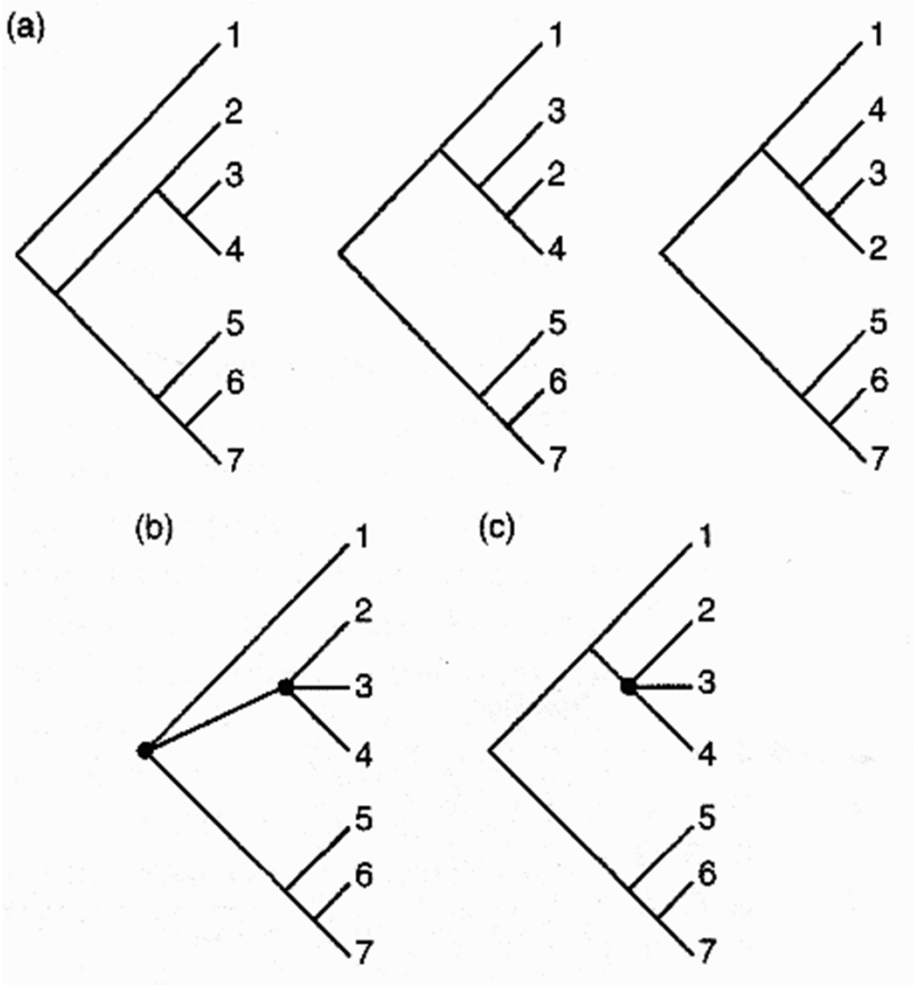
- Metody založené na znacích využívají pro konstrukci stromu přímo matici vícenásobného zarovnání
- Hledání stromu rozděleno na malý a velký parsinomy problém
- Malý parsinomy problém: strom je již vytvořen, hledá se vnitřní ohodnocení uzlů – Fitchův a Sarkoffův algoritmus s O(kn)
- Velký parsinomy problém: zahrnuje i nalezení vhodného stromu – obecně NP-úplný problém, nutno použít heuristiku
- Zavedení relace sousedících stromů, základní prohledávání sousedních stromů ale obvykle končí v lokálním minimu => potřeba zavést složitější transformace založené na prořezávání a přeuspořádání stromu

Porovnání metod

- Nelze obecně říci, že metody založené na vzdálenosti jsou lepší než metody založené na znacích
- Existuje velké množství variant těchto přístupů - většina z nich pracuje správně
- Problémy nastávají pokud:
  - mezi sekvencemi je velké množství změn
  - neuvažuje se různá rychlost substitucí v oddělených větvích
- Často se aplikuje více metod na vstupní množinu sekvencí
  - Každá metoda vytvoří svůj strom
  - Z jednotlivých stromů se pak vytváří výsledný tzv. Consensus Tree

Consensus Tree

- Používá se i v případě, že několik stromů má stejnou úroveň parsimony
- Obecné pravidlo:
  - větve, které jsou shodné ve všech stromech jsou umístěny i ve výsledném stromu (jako rozdvojené)
  - větve, které se neshodují se spojují do n-ární uzlů
- Striktní consensus tree = všechny neshody jsou ohodnoceny stejně
- 50% majority consensus tree = pokud pro danou neshodu platí, že ve více než polovině stromů se shoduje, potom se použije běžné rozvětvení

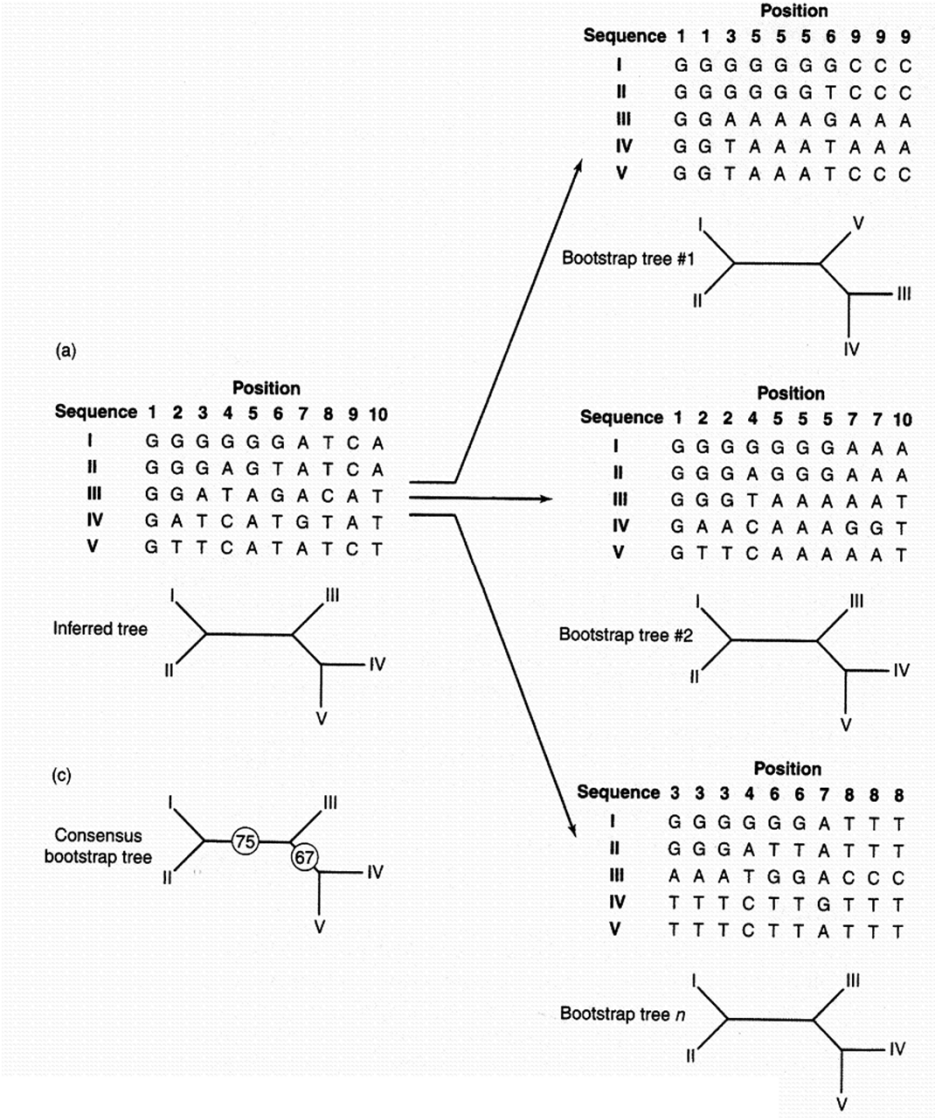


Důvěryhodnost stromu

- Všechny vytvořené stromy reprezentují pouze hypotézu o evoluční historii skupiny sekvencí
- Jako u každé hypotézy je potřeba se zeptat:
  - Jaká je míra důvěryhodnosti stromu a všech jeho částí?
  - Jaká je míra pravděpodobnosti, že daný strom je korektnější než jiný nebo náhodně vygenerovaný?
- Řešení:
  - Bootstrapping + Consensus tree

Bootstrapping

- Postup:
  1. K dané množině sekvencí se sestrojí strom
  2. Množina sekvencí se nastříhá po sloupcích a vloží se do tašky
  3. Z kabely se náhodně vybírají nastříhané sloupce a z nich se sestavují nové sady sekvencí stejné délky (některé sloupce se mohou vyskytovat více-krát)
  4. Pro každou novou sadu sekvencí se sestrojí strom
  5. Pro výslednou skupinu stromů se sestrojí consensus strom, kde jednotlivé větve jsou ohodnoceny mírou shodu přes všechny odvozené stromy
- Problémy:
  - je potřeba udělat několik set iterací, jinak není metoda důvěryhodná - existují vylepšené varianty



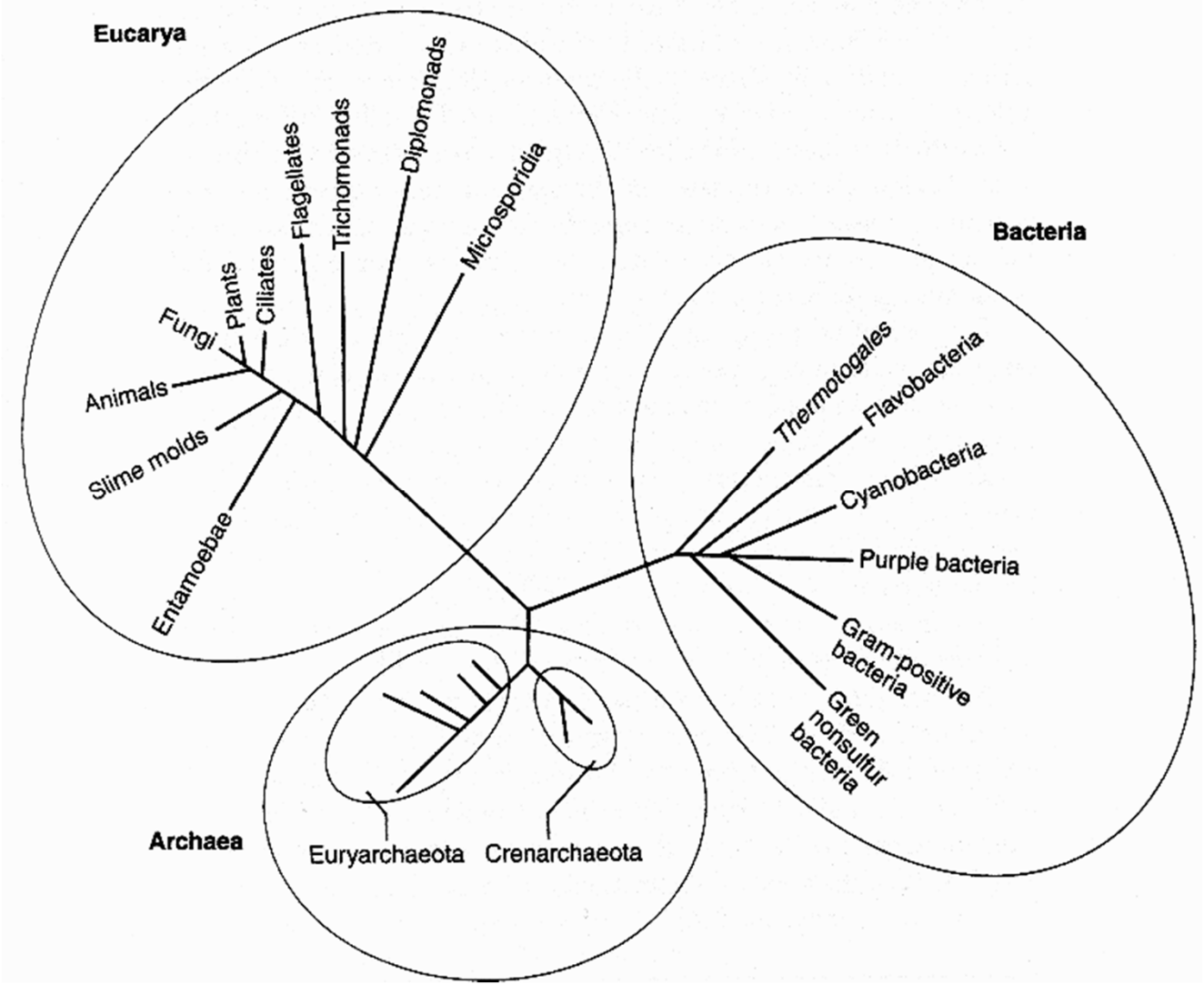
Strom života

- Jednou ze základních otázek, která trápila biology po 10-letí byl vznik a rozdělení živých organizmů na Zemi - sestavení stromu života
- Zpočátku biologové rozdělovali organizmy toho, jak vypadají a jakou mají strukturu (fenotyp), typicky na rostliny a zvířata
- Následně, byly rozdělovány na základě struktury buňky na prokaryoty a eukaryoty (obsahují buněčné jádro)
- Následně bylo rozdělení rozšířeno na 5 skupin podle absence buněčné membrány na: prokaryoty, protisty (jednobuněčné organizmy), rostliny, ryby a zvířata - opět špatně

Strom života

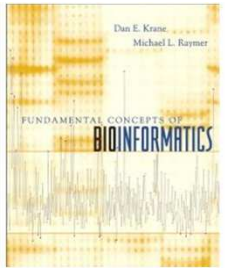
- V 70-letech byly poprvé k dispozici první sekvence RNA a DNA
- Carl Woese a jeho kolegové zkonstruovali strom života z 16 vzorků RNA různých organizmů
- Strom je rozdělen do tří skupin:
  - bakterie (prokaryoty, mitochondrie a chloroplasty)
  - eukarya (rostliny, zvířata a ryby)
  - archaea (teplomilné bakterie a ostatní málo známe organizmy)
- Toto rozdělení nebylo zjistitelné z žádných fenotypových vlastností ani zkamenělin
- Doposud nejpodporovanější struktura

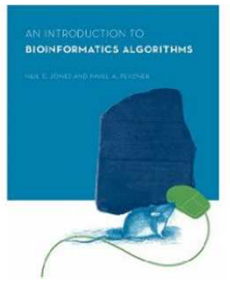
Strom života



Literatura

- Dan K. Krane, Michael L. Raymer: ***Fundamental Concepts of Bioinformatics***, ISBN: 0-8053-4633-3, Benjamin Cummings 2003.
- Neil C. Jones, Pavel A. Pevzner, ***An Introduction to Bioinformatics Algorithms***, ISBN-10: 0262101068, The MIT Press, 2004





Konec

Děkuji za pozornost