Topics of today:

1. Small Parsimony / Big Parsimony

2. Fitch's algorithm

3. Small Parsimony with SCJ

4. Quiz-review
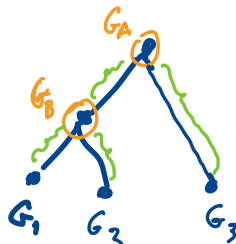
Small (and big) parsimony problems

$d_M(\mathbb{G}_1, \mathbb{G}_2)$: distance between two genomes under some model M

Binary tree T $\begin{cases} \text{topology of T is known} \\ \text{each leaf u of T corresponds to a given genome } \mathbb{G}_u \end{cases}$

Function $\mathcal{A}$: assigns a genome $\mathcal{A}(u)$ to each node u of T
$\qquad$ (if u is a leaf of T, then $\mathcal{A}(u) = \mathbb{G}_u$)

Weight $w(uv)$ of a branch uv of T under $\begin{cases} \text{assignment } \mathcal{A} \\ \text{model } M \end{cases}$ $: w(uv) = d_M(\mathcal{A}(u), \mathcal{A}(v))$

Small parsimony problem under the model M:

$\qquad$ Find an assignment $\mathcal{A}$ that minimizes the total branch weight of T:

$$\mathcal{W}(T) = \min_{\mathcal{A} \in \mathfrak{A}} \sum_{uv \in E(T)} d_M(\mathcal{A}(u), \mathcal{A}(v))$$

Big parsimony problem under the model M:

$\qquad$ Given a set of k genomes $\mathbb{G}_1, \mathbb{G}_2, ..., \mathbb{G}_k$,

$\qquad\qquad$ find $\begin{cases} \text{tree T whose } k \text{ leaves are in one-to-one correspondence with the genomes } \mathbb{G}_1, \mathbb{G}_2, ..., \mathbb{G}_k \\ \text{assignment } \mathcal{A} \text{ of genomes to the nodes of T} \end{cases}$

$\qquad$ minimizing the total branch weight $\mathcal{W}(T)$

Small parsimony with Fitch's algorithm

General model:
$$\begin{cases} \text{binary rooted tree T, with } n \text{ nodes} \\ \text{genomes are represented by sequences of a fixed length } \ell, \text{over a finite alphabet } \Sigma \text{ (with } |\Sigma| = m) \\ \text{hamming distance (hd) gives the weight of the branches} \end{cases}$$

Assuming positions being mutually independent, the problem can be solved for each position separately:
The algorithm determines an optimal value for each position p of each node v, denoted by $s_v[p]$

Bottom-up phase: defines set $B(v, p)$ of possible values for each $s_v[p]$, based on v's children x_1 and x_2
$$\begin{cases} \text{if } v \text{ is a leaf}: B(v, p) = \{s_v[p]\} \\ \text{else} \begin{cases} \text{compute } B(v, p) = B(x_1, p) \cap B(x_2, p) \\ \text{if } B(v, p) = \emptyset : B(v, p) = B(x_1, p) \cup B(x_2, p) \end{cases} \end{cases}$$
Complexity: $O(mn)$ (bottom-up traversal takes $O(n)$, computation of each node takes $O(m)$)

Top-down phase: defines final value $s_v[p]$, based on set $B(v, p)$ and v's parent u
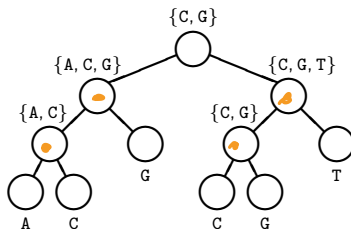$$\begin{cases} \text{if } v \text{ has a parent } u \text{ and } s_u[p] \in B(v, p) : s_v[p] = s_u[p] \\ \text{otherwise (including the root), arbitrarily assign any value from } B(v, p) \text{ to } s_v[p] \end{cases}$$
Complexity: $O(mn)$ (top-down traversal takes $O(n)$, computation of each node takes $O(m)$)
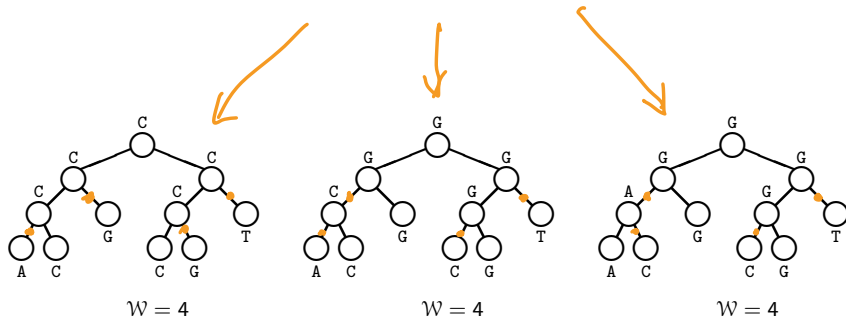
Total complexity: $O(mn)$ per position; with ℓ positions: $O(\ell mn)$

Small parsimony with Fitch's algorithm

Bottom-up phase:

Top-down phase:

$\mathcal{W} = 4$

$\mathcal{W} = 4$

$\mathcal{W} = 4$

Small parsimony under SCJ with Fitch's algorithm

$\mathcal{F} = \{1, 2\}$

$\widehat{\alpha}(\mathcal{F}) = \{1^h 1^t, 1^t 2^h, 1^h 2^t,$
$\qquad\qquad 1^t 2^h, 1^t 2^t,$
$\qquad\qquad 2^h, 2^t \}$

$\begin{cases}
\text{set of gene families } \mathcal{F} \\[2mm]
\text{set of extremities } \xi(\mathcal{F}) = \{f^t : f \in \mathcal{F}\} \cup \{f^h : f \in \mathcal{F}\} \quad (|\xi(\mathcal{F})| = 2|\mathcal{F}|) \\[2mm]
\text{set of all possible adjacencies } \widehat{\alpha}(\mathcal{F}) = \{\text{subsets of } \xi(\mathcal{F}) \text{ with size 2}\} \\
\qquad\qquad \Rightarrow \ |\widehat{\alpha}(\mathcal{F})| = \binom{|\xi(\mathcal{F})|}{2} = \binom{2|\mathcal{F}|}{2} = \frac{2|\mathcal{F}|(2|\mathcal{F}|-1)}{2} = O(|\mathcal{F}|^2) \\
\qquad\qquad \text{Note: } \widehat{\alpha}(\mathcal{F}) \text{ has many pairs of } \textbf{conflicting adjacencies } xy, xz \text{ with } y \neq z
\end{cases}$

genome assigned
to vertex u
$\begin{cases}
\text{represented by a sequence } s_u \text{ of length } \ell = |\widehat{\alpha}(\mathcal{F})| \text{ over the binary alphabet } \{0, 1\} \\[2mm]
\text{cannot contain conflicting adjacencies:} \\
\qquad \text{for each pair of conflicting adj. in } \widehat{\alpha}(\mathcal{F}), \text{ at most one of the two can be set to 1 in } s_u
\end{cases}$

Small parsimony under SCJ with Fitch's algorithm

xy at position p_1 xz at position p_2

(a)

$B(v, p_1)$ $\{1\}$ $\Rightarrow$ $B(v, p_2)$ $\{0\}$

$\{1\}$ $\{1\}$ $\{0\}$ $\{0\}$

(b)

$B(v, p_1)$ $\{1\}$ $\Rightarrow$ $B(v, p_2)$ $\{0\}$

$\{1\}$ $\{0, 1\}$ $\{0\}$ $\{0\}$ or $\{0, 1\}$

Possible bottom-up scenarios involving two conflicting adjacencies
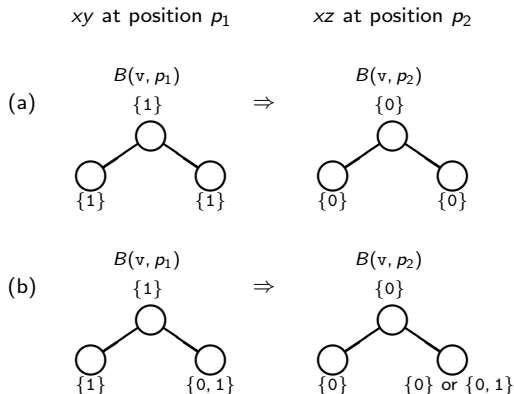
Conflicts can be avoided by assigning the value 0 to each ambiguous position of the root genome:

$\mathtt{r}$ is the root of $\mathtt{T}$: for each position p of $s_\mathtt{r}$ $\begin{cases} s_\mathtt{r}[p] = 1 & \text{if } B(\mathtt{r}, p) = \{1\} \\ s_\mathtt{r}[p] = 0 & \text{otherwise} \end{cases}$

It is easy to verify that: $d_{\mathrm{SCJ}}(\mathtt{u}, \mathtt{v}) = \mathrm{hd}(s_\mathtt{u}, s_\mathtt{v})$

Close-related problems are NP-hard...

Small parsimony under breakpoint distance

Big parsimony under SCJ

References

Toward Defining the Course of Evolution: Minimum Change for a Specific Tree Topology

(Walter M. Fitch)

Systematic Zoology, vol. 20, pp. 406–416 (1971)

SCJ: A Breakpoint-Like Distance that Simplifies Several Rearrangement Problems

(Pedro Feijão and João Meidanis)

TCBB volume 8 Number: 5 (2011)

Quiz – Review

Inversion model 1

1 In the DCJ model any operation reconstructing a target adjacency is optimal, but the same is not true for the inversion model because...

    A a target adjacency can be bad

    B a target adjacency can be already present in the genome

    C reconstructing a target adjacency can be unsafe

2 A cycle is bad when...

    A it cannot be sorted by inversions

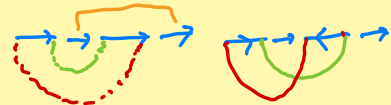   B it interleaves another bad cycle

    C it contains only bad target adjacencies

3 Which data structure helps finding safe inversions?

    A relational diagram

    B overlap graph

    C component tree

4 A bad component can be fixed...

    A with a neutral inversion

    B with a split inversion
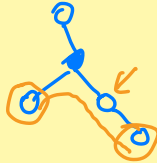
    C with a safe inversion

Inversion model 2

1 Each leaf of the component tree represents...

    A   a bad component

    B   a hurdle

    C   a fortress

2 The cost of covering a component tree can be expressed in terms of...

    A   the number of bad nodes

    B   the length of the longest traversal of the tree

    C   the number of leaves

3 Fixing a super hurdle with a neutral inversion

    A   is a good strategy

    B   creates a new hurdle

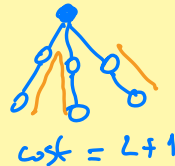   C   destroys a good component

4 Merging two good (or trivial) components..

    A   can merge bad components into a good one

    B   creates a new bad component

    C   is never recommended

cost : L

cost = L + 1

DCJ-indel model 1

1   The indel-potential is defined as...

   A   the number of runs in a component

   B   the smallest number of runs that can be obtained after sorting with internal gaining DCJs

   C   the number of indel-edges in a component

2   The indel-potential of a component depends on..

   A   its number of runs

   B   its number of indel-edges

   C   its length

3   The number of runs in a cycle can be...

   A   0,1,2,4,6,8,...

   B   any non-negative integer

   C   any positive integer

4   The number of runs in a path can be...

   A   0,1,2,4,6,8,...

   B   any non-negative integer   0,1,2,3,4,5 ...

   C   any positive integer   1,2,3,4,5 ...
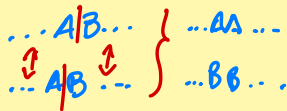
DCJ-indel model 2

1 A recombination can reduce the overall number of runs by at most...

    A 1   **B 2**   C 3

2 A recombination can reduce the overall overall indel-potential by at most...

    A 1   **B 2**   C 3

3 A recombination involving a cycle is...

    A gaining

    B neutral

    C losing

4 A recombination involving a cycle can be...

    A deducting

    B part of an optimal sorting sequence if $\Delta_\lambda = -2$

    C none of those

$$\left. \begin{array}{l} \dots A|B \dots \\ \dots A|B \dots \end{array} \right\} \begin{array}{l} \dots AA \dots \\ \dots BB \dots \end{array}$$

Λ	λ
0	0
1	1
2	2
3	2
4	3
5	3

DCJ-indel model - Path recombinations

With respect to the endpoints:

$\text{A}{+}\text{A} \ + \ \mathbb{B}{+}\mathbb{B}$
$\begin{cases} \text{A}-\mathbb{B} \ + \ \text{A}-\mathbb{B} & \text{(gaining)} \\ \text{A}-\mathbb{B} \ + \ \text{A}-\mathbb{B} & \text{(gaining)} \end{cases}$

$\text{A}{+}\mathbb{B} \ + \ \text{A}{+}\mathbb{B}$
$\begin{cases} \cancel{\text{A}-\text{A} \ + \ \mathbb{B}-\mathbb{B} \quad \text{(losing)}} \\ \text{A}-\mathbb{B} \ + \ \text{A}-\mathbb{B} & \text{(neutral)} \end{cases}$

$\text{A}{+}\text{A} \ + \ \text{A}{+}\text{A}$
$\begin{cases} \text{A}-\text{A} \ + \ \text{A}-\text{A} & \text{(neutral)} \\ \text{A}-\text{A} \ + \ \text{A}-\text{A} & \text{(neutral)} \end{cases}$

With respect to the runs:

$\mathcal{AB} \ + \ \mathcal{AB}$
$\begin{cases} \mathcal{BAB} + \varepsilon & (\Delta_\lambda = -2) \\ \mathcal{AA} \ + \ \mathcal{BB} & (\Delta_\lambda = -2) \\ \mathcal{ABBA} \ + \ \varepsilon & (\Delta_\lambda = -2) \end{cases}$

$\mathcal{A(B)} \ + \ \mathcal{A}$
$\begin{cases} \mathcal{AA} \ + \ (\mathcal{B}) & (\Delta_\lambda = -1) \\ \mathcal{AA(B)} \ + \ \varepsilon & (\Delta_\lambda = -1) \end{cases}$

$(\mathcal{A})\mathcal{B} \ + \ \mathcal{B}$
$\begin{cases} (\mathcal{A}) \ + \ \mathcal{BB} & (\Delta_\lambda = -1) \\ (\mathcal{A})\mathcal{BB} \ + \ \varepsilon & (\Delta_\lambda = -1) \end{cases}$

Deducting path recombinations:
$\begin{cases} \text{gaining with } \Delta_\lambda = -2 \\ \text{gaining with } \Delta_\lambda = -1 \\ \text{neutral with } \Delta_\lambda = -2 \end{cases}$

DCJ-indel model - Path recombinations

Putting together (examples):

$$\mathbb{AB}_{\mathcal{AB}}$$
$$\mathbb{A}-\mathcal{AB}-\mathbb{B} \quad + \quad \mathbb{A}-\mathcal{AB}-\mathbb{B} \quad = \quad \mathbb{A}-\mathcal{ABAB}-\mathbb{B} \quad + \quad \mathbb{A}-\mathbb{B} \qquad \text{(neutral, with } \Delta_\lambda = -1\text{)}$$

$$\mathbb{A}-\mathcal{AB}-\mathbb{B} \quad + \quad \mathbb{A}-\mathcal{AB}-\mathbb{B} \quad = \quad \mathbb{A}-\mathcal{ABAB}-\mathbb{B} \quad + \quad \mathbb{A}-\mathbb{B} \qquad \text{(neutral, with } \Delta_\lambda = -1\text{)}$$

$$\mathbb{A}-\mathcal{AB}-\mathbb{B} \quad + \quad \mathbb{A}-\mathcal{BA}-\mathbb{B} \quad = \quad \mathbb{A}-\mathcal{ABBA}-\mathbb{B} \quad + \quad \mathbb{A}-\mathbb{B} \qquad \text{(neutral, with } \Delta_\lambda = -2\text{)}$$

$$\mathbb{A}-\mathcal{AB}-\mathbb{B} \quad + \quad \mathbb{A}-\mathcal{BA}-\mathbb{B} \quad = \quad \mathbb{A}-\mathcal{AA}-\mathbb{B} \quad + \quad \mathbb{A}-\mathcal{BB}-\mathbb{B} \qquad \text{(neutral, with } \Delta_\lambda = -2\text{)}$$

$$\mathbb{A}-\mathcal{AB}-\mathbb{B} \quad + \quad \mathbb{A}-\mathcal{BA}-\mathbb{B} \quad = \quad \mathbb{A}-\mathcal{BAAB}-\mathbb{B} \quad + \quad \mathbb{A}-\mathbb{B} \qquad \text{(neutral, with } \Delta_\lambda = -2\text{)}$$

$$\mathbb{A}-\mathcal{AB}-\mathbb{B} \quad + \quad \mathbb{A}-\mathcal{BA}-\mathbb{B} \quad = \quad \mathbb{A}-\mathcal{BB}-\mathbb{B} \quad + \quad \mathbb{A}-\mathcal{AA}-\mathbb{B} \qquad \text{(neutral, with } \Delta_\lambda = -2\text{)}$$

Λ	λ
0	0
1	1
2	2
3	2
4	3
5	3
6	4
7	4

DCJ-indel model - Path recombinations

Putting together (examples):

$\mathbb{AB}_{\mathcal{AB}}$
$\mathbb{A}—\mathcal{AB}—\mathbb{B}$ + $\mathbb{AB}_{\mathcal{BA}}$ $\mathbb{A}—\mathcal{BA}—\mathbb{B}$ = $\mathbb{AB}_{\mathcal{A}}$ $\mathbb{A}—\mathcal{ABBA}—\mathbb{B}$ + $\mathbb{AB}_{\varepsilon}$ $\mathbb{A}—\mathbb{B}$ (neutral, with $\Delta_{\lambda} = -2$)

2   2 (4)   3 (2)

$\mathbb{AB}_{\mathcal{AB}}$
$\mathbb{A}—\mathcal{ABAB}—\mathbb{B}$ + $\mathbb{AB}_{\mathcal{BA}}$ $\mathbb{A}—\mathcal{BA}—\mathbb{B}$ = $\mathbb{AB}_{\mathcal{A}}$ $\mathbb{A}—\mathcal{ABABBA}—\mathbb{B}$ + $\mathbb{AB}_{\varepsilon}$ $\mathbb{A}—\mathbb{B}$ (neutral, with $\Delta_{\lambda} = -2$)

4   2 (5)   5 (3)

$\mathbb{AB}_{\mathcal{AB}}$
$\mathbb{A}—\mathcal{ABAB}—\mathbb{B}$ + $\mathbb{AB}_{\mathcal{BA}}$ $\mathbb{A}—\mathcal{BABA}—\mathbb{B}$ = $\mathbb{AB}_{\mathcal{A}}$ $\mathbb{A}—\mathcal{ABABBABA}—\mathbb{B}$ + $\mathbb{AB}_{\varepsilon}$ $\mathbb{A}—\mathbb{B}$ (neutral, with $\Delta_{\lambda} = -2$)

4   4 (6)   7 (4)

| Λ	λ
0	0
1	1
2	2
3	2
4	3
5	3
6	4
7	4

DCJ and DCJ-indel models - Capping

Add caps to close all paths of the graph into cycles, preserving the distance

Canonical capping (no indel edges): maximizes the number of cycles

paths	linking cycle	Δn	Δc	$\Delta(2\mathbb{AB})$	Δ_{DCJ}
$\mathbb{AB}$	$(\mathbb{AB})$	+0.5	+1	−0.5	0
$\mathbb{AA} + \mathbb{BB}$	$(\mathbb{AA}, \mathbb{BB})$	+1	+1	0	0
$\mathbb{AA}$	$(\mathbb{AA}, \Gamma_{\mathbb{B}})$	+1	+1	0	0
$\mathbb{BB}$	$(\mathbb{BB}, \Gamma_{\mathbb{A}})$	+1	+1	0	0

Singular capping (with indel edges): optimizes the number of cycles and of runs at the same time
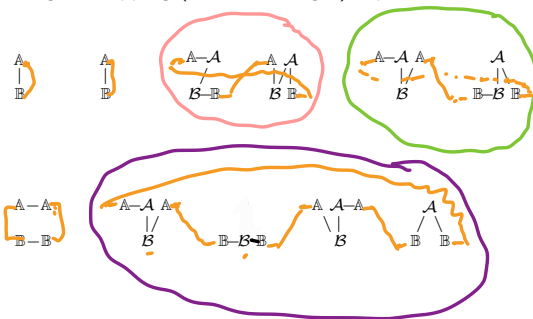
paths	linking cycle	Δn	Δc	$\Delta(2\mathbb{AB})$	$\Delta\lambda$	Δ^{λ}_{DCJ}
$\mathbb{AA}_{\mathcal{AB}} + \mathbb{BB}_{\mathcal{AB}}$	$(\mathbb{AA}_{\mathcal{AB}}, \mathbb{BB}_{\mathcal{BA}})$	+1	+1	0	−2	−2
$2 \times \mathbb{AA}_{\mathcal{AB}} + \mathbb{BB}_{\mathcal{A}} + \mathbb{BB}_{\mathcal{B}}$	$(\mathbb{AA}_{\mathcal{AB}}, \mathbb{BB}_{\mathcal{B}}, \mathbb{AA}_{\mathcal{BA}}, \mathbb{BB}_{\mathcal{A}})$	+2	+1	0	−4	−3
$\mathbb{AB}_{\mathcal{AB}} + \mathbb{AB}_{\mathcal{BA}}$	$(\mathbb{AB}_{\mathcal{AB}}, \mathbb{AB}_{\mathcal{BA}})$	+1	+1	−1	−2	−1
$\mathbb{AB}$	$(\mathbb{AB})$	+0.5	+1	−0.5	0	0
$\mathbb{AA} + \mathbb{BB}$	$(\mathbb{AA}, \mathbb{BB})$	+1	+1	0	0	0