

Quartet distance between general trees

Chris Christiansen

Thomas Mailund

Christian N.S. Pedersen

Martin Randers



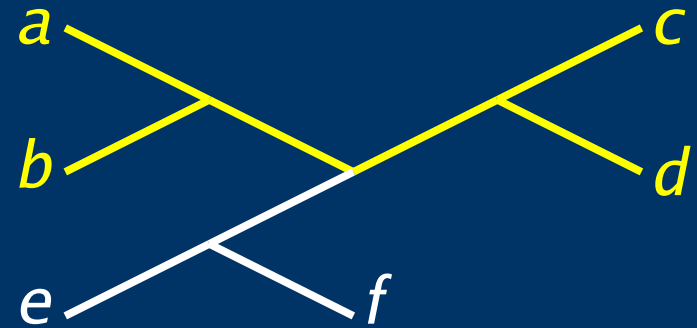
Evolutionary trees

- An *evolutionary tree* describes relationship for a set of species
- Can be constructed using different traits → slightly different trees
- Wish to compare these
- This talk: *Quartet distance* between trees over same set of n species, S

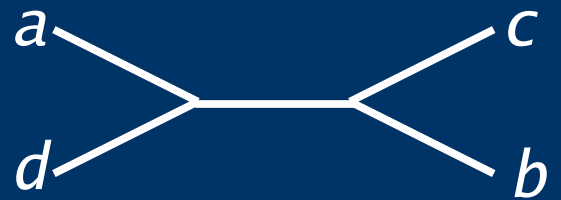
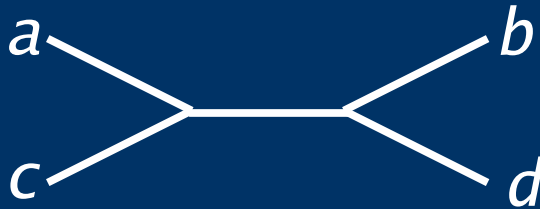
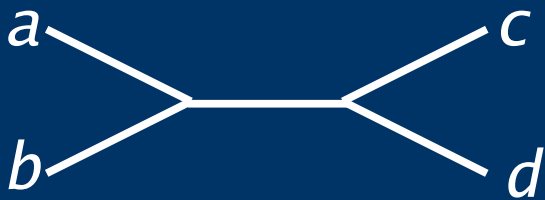


Quartets and quartet topologies

- A *quartet* is a sub-set of species of size 4, e.g. $\{a,b,c,d\} \subseteq S$
- The *quartet topology* of a quartet $\{a,b,c,d\}$ is the induced subtree



Possible topologies of {a,b,c,d}



Butterfly topologies

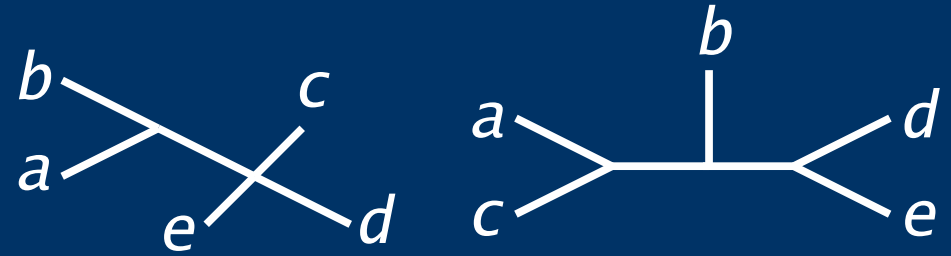


Star topology



Quartet distance

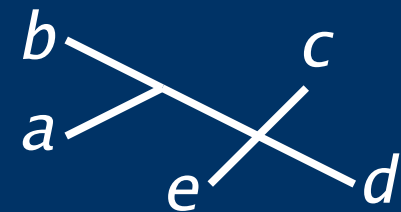
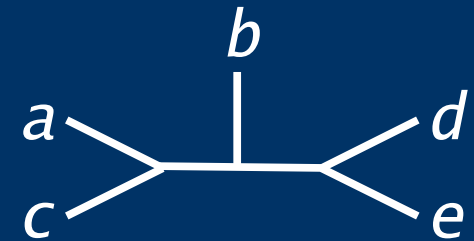
- The **quartet distance** between T_1 and T_2 is the number of quartets, $\{a, b, c, d\}$, where the topology differ



- $ab|cd \neq ac|bd$
- $ab|ce \neq ac|be$
- $ab|de = ab|de$
- $ac \times ed \neq ac|de$
- $bc \times ed \neq bc|de$
- $qdist(T_1, T_2) = 4$

Binary trees and general trees

- Results for **binary trees**:
 - Steel and Penny (1993): $O(n^3)$
 - Bryant *et al.* (2000): $O(n^2)$
 - Brodal *et al.* (2003): $O(n \log n)$
 - n number of leaves
- For **arbitrary degree**:
 - This paper: $O(n^3)$ and $O(n^2 d^2)$
 - n number of leaves
 - d maximal degree



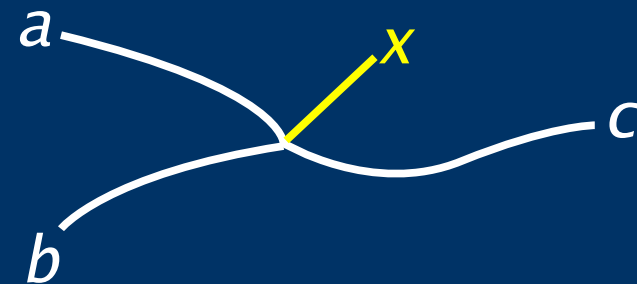
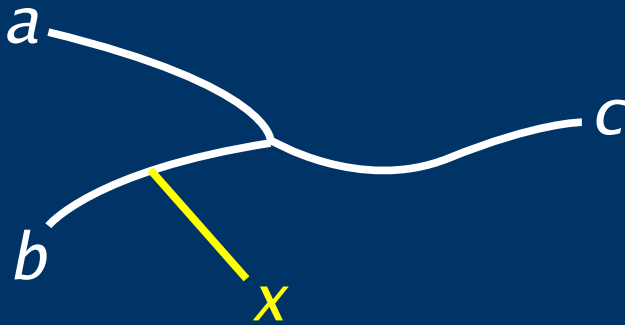
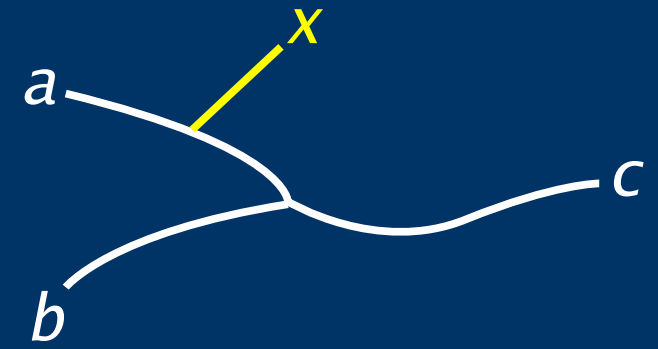
A triplet approach

- A *triplet based* approach:
 - For each *fixed triplet* $\{a,b,c\}$



A triplet approach

- A **triplet based** approach:
 - For each **fixed triplet** $\{a,b,c\}$
 - And each leaf x in $S-\{a,b,c\}$



Shared topologies per triplet

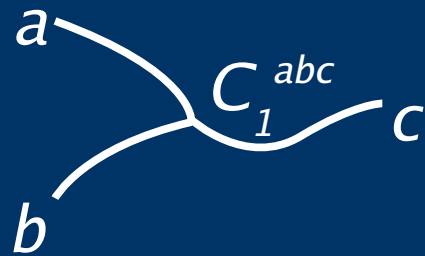
- A *triplet based* approach:
 - For each *fixed triplet* $\{a,b,c\}$
 - And each leaf x in $S-\{a,b,c\}$
 - Compute *shared quartet topologies* $\{a,b,c,x\}$
 - The sum of these, for each triplet $\{a,b,c\}$, divided by 4 is *all shared topologies*



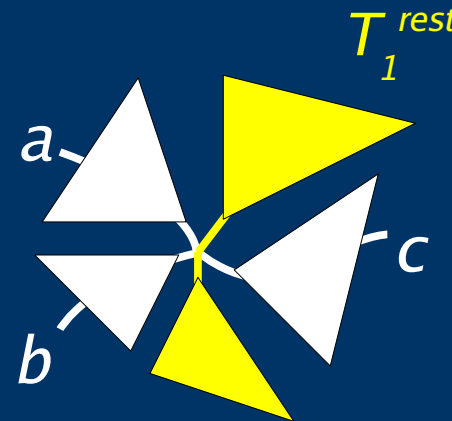
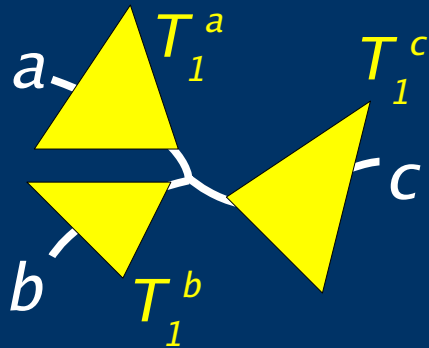
$$qdist(T_1, T_2) = \binom{n}{4} - \frac{1}{4} \sum_{a,b,c} \text{shared}(a,b,c)$$

Triplet centers

- Consider the **triplet centers**, C_1^{abc} and C_2^{abc}
 - The “meeting point” of the paths between a , b , and c



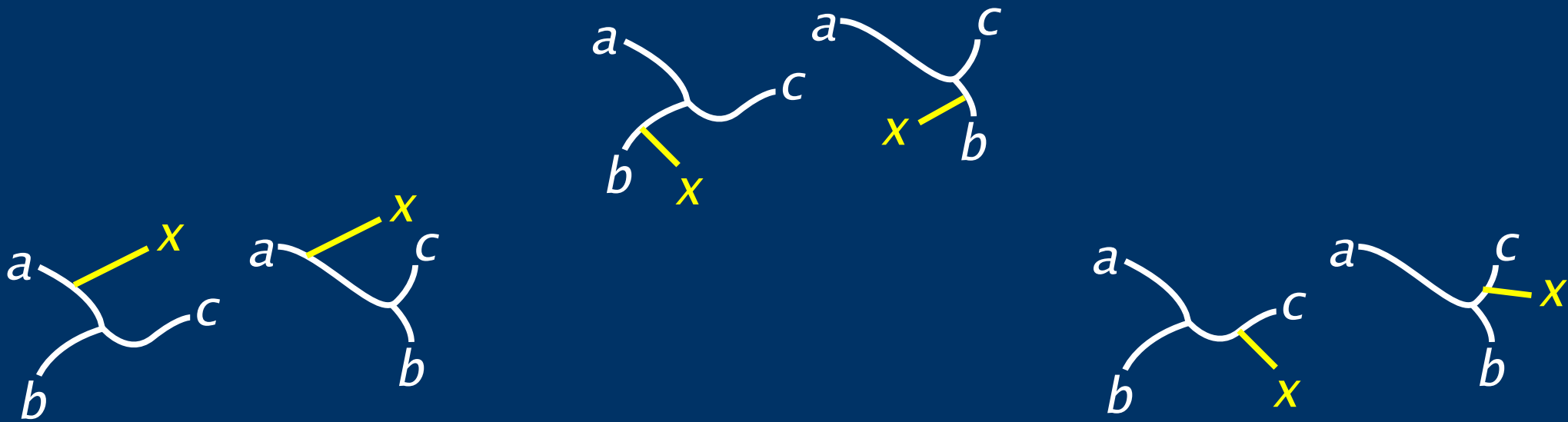
- Induced subtrees:



Shared butterfly topologies

- Shared butterfly topologies:

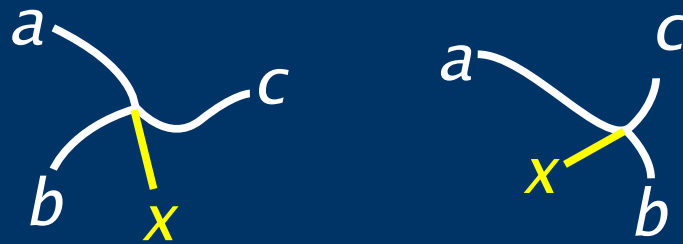
$$|\mathcal{T}_1^a \cap \mathcal{T}_2^a| - 1 + |\mathcal{T}_1^b \cap \mathcal{T}_2^b| - 1 + |\mathcal{T}_1^c \cap \mathcal{T}_2^c| - 1$$



Shared star topologies

- Shared star topologies:

$$|\mathcal{T}_1^{rest} \cap \mathcal{T}_2^{rest}|$$



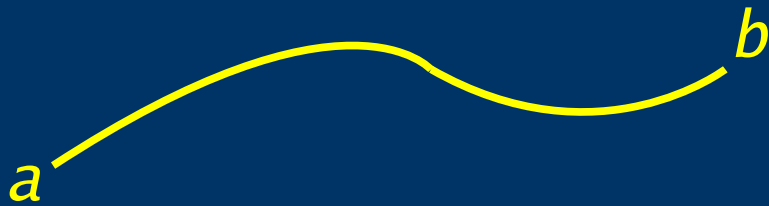
- With a bit of algebra, expressible through sizes and intersections of T_i^a , T_i^b , T_i^c , $i=1,2$
-
-

Preprocessing

- As a *preprocessing step*
 - Tabulate all $|T_i^x|$ and $|T_1^x \cap T_2^y|$
 - Time and space $O(n^2)$ – Bryant *et al.* (2000)
- Shared quartet topologies of $\{a,b,c,x\}$
(fixed a,b,c) computable in $O(1)$

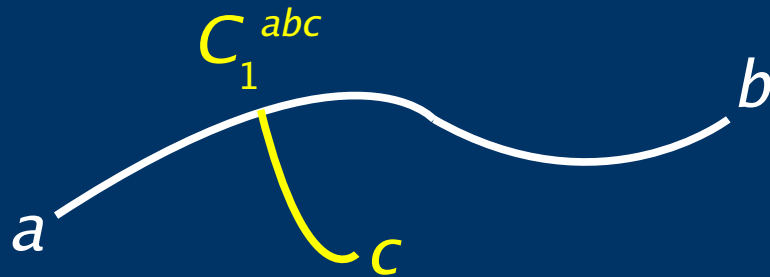
The triplet algorithm

- For each pair a, b :
 - Calculate *the paths between a and b*

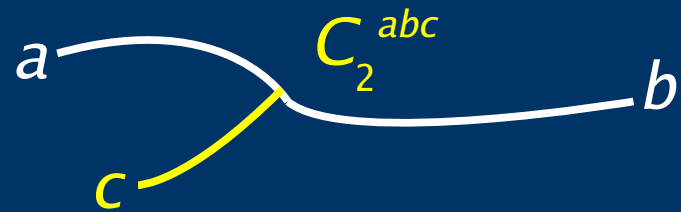


The triplet algorithm

- For each pair a, b :
 - Calculate the paths between a and b
 - *Tabulate C_i^{abc} for each c*



$$C_1[c] := C_1^{abc}$$



$$C_2[c] := C_2^{abc}$$

The triplet algorithm

- For each pair a, b :
 - Calculate the paths between a and b
 - Tabulate C_i^{abc} for each c
 - *Calculate $shared(a, b)$ as*

$$shared(a, b) = \sum_c shared(a, b, c)$$

Time complexity

- For each pair a, b :
 - Calculate the paths between a and b
 - Tabulate C_i^{abc} for each c
 - Calculate $\text{shared}(a, b)$ as

$$\text{shared}(a, b) = \sum_c \text{shared}(a, b, c)$$

Four tree traversals: $O(n)$

Time complexity

- For each pair a, b :
 - Calculate the paths between a and b
 - Tabulate C_i^{abc} for each c
 - Calculate $\text{shared}(a, b)$ as

$$\text{shared}(a, b) = \sum_c \text{shared}(a, b, c)$$

Four tree traversals: $O(n)$

n constant time summations: $O(n)$

Time complexity: $O(n^3)$

- For each pair a, b :
 - Calculate the paths between a and b
 - Tabulate C_i^{abc} for each c
 - Calculate $\text{shared}(a, b)$ as

$$\text{shared}(a, b) = \sum_c \text{shared}(a, b, c)$$

Four tree traversals: $O(n)$

n constant time summations: $O(n)$

Repeat n^2 times: Total $O(n^3)$

Reducing to butterflies

- Notation:

- $S(T_1, T_2)$ = |one tree has star topology|
- $\text{shared}_B(T_1, T_2)$ = |equal butterfly topology|
- $\text{diff}_B(T_1, T_2)$ = |different butterfly topology|

	S_1	S_2	B_1	B_2
$\{a, b, c, d\}$	1	1	0	0
$\{a, b, c, e\}$	1	0	0	1
$\{a, b, c, f\}$	0	0	1	1
...				
$\{x, y, z, w\}$	0	1	1	0

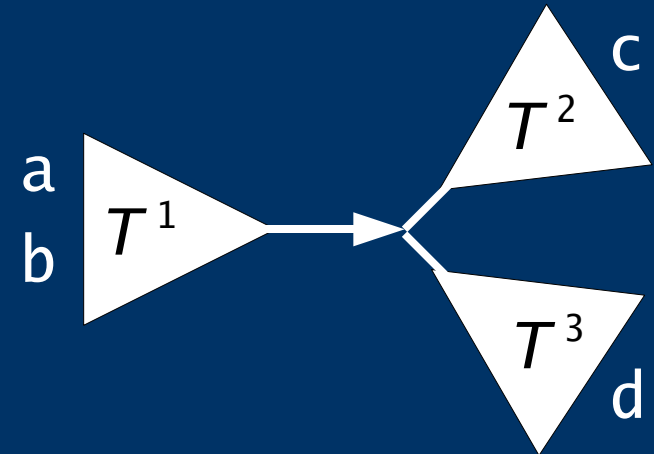
- Indirect computation:

- $\text{qdist}(T_1, T_2) = S(T_1, T_2) + \text{diff}_B(T_1, T_2)$
 $= B_1 + B_2 - 2 \cdot \text{shared}_B(T_1, T_2) - \text{diff}_B(T_1, T_2)$
- $B_i = \text{shared}_B(T_i, T_i)$,
- $\text{shared}_B(T_1, T_2)$ and $\text{diff}_B(T_1, T_2)$ computed in a similar way

$$S(T_1, T_2) = B_1 + B_2 - 2 \cdot (\text{shared}_B(T_1, T_2) + \text{diff}_B(T_1, T_2))$$

Claims and butterflies

- A *claim* is an *oriented split* in three trees
 - Defines a set of butterflies: $ab|cd$
 - Each butterfly claimed by exactly two claims
- A *pair of claims* share butterflies:



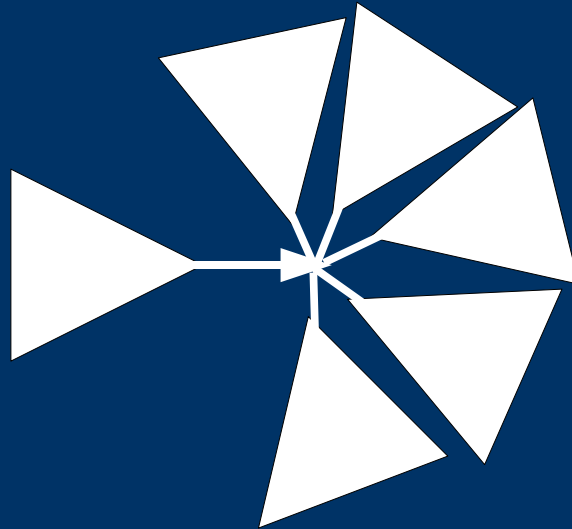
$$\text{count}(cl_1, cl_2) = \binom{|T_1^1 \cap T_2^1|}{2} \cdot (|T_1^2 \cap T_2^2| \cdot |T_1^3 \cap T_2^3| + |T_1^2 \cap T_2^3| \cdot |T_1^3 \cap T_2^2|)$$

- *Shared butterflies*:

$$\text{shared}_B(T_1, T_2) = \frac{1}{2} \sum_{cl_1 \in T_1} \sum_{cl_2 \in T_2} \text{count}(cl_1, cl_2)$$

High-degree nodes

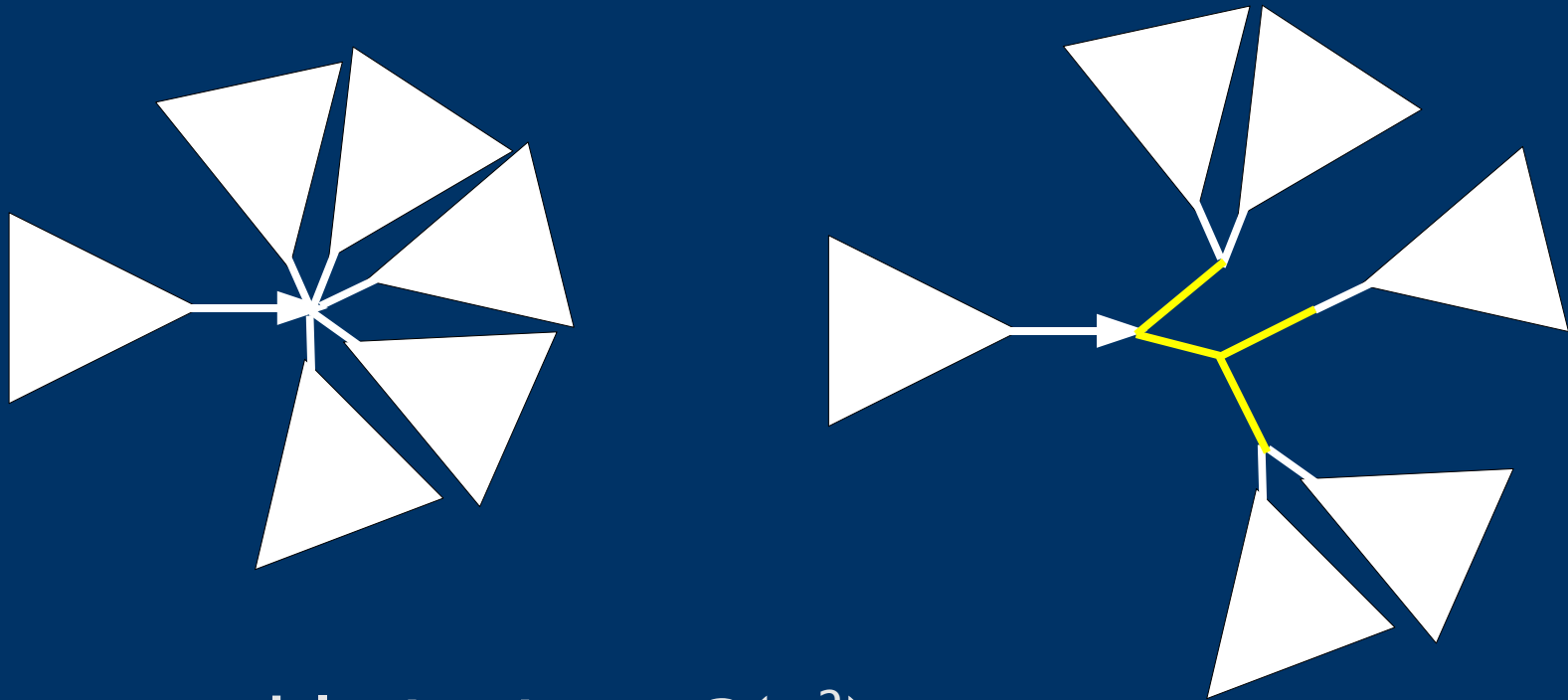
- A node of degree d has $(d-1)(d-2)/2$ claims per ingoing edge



- $O(n^2 d^4)$ pairs of claims



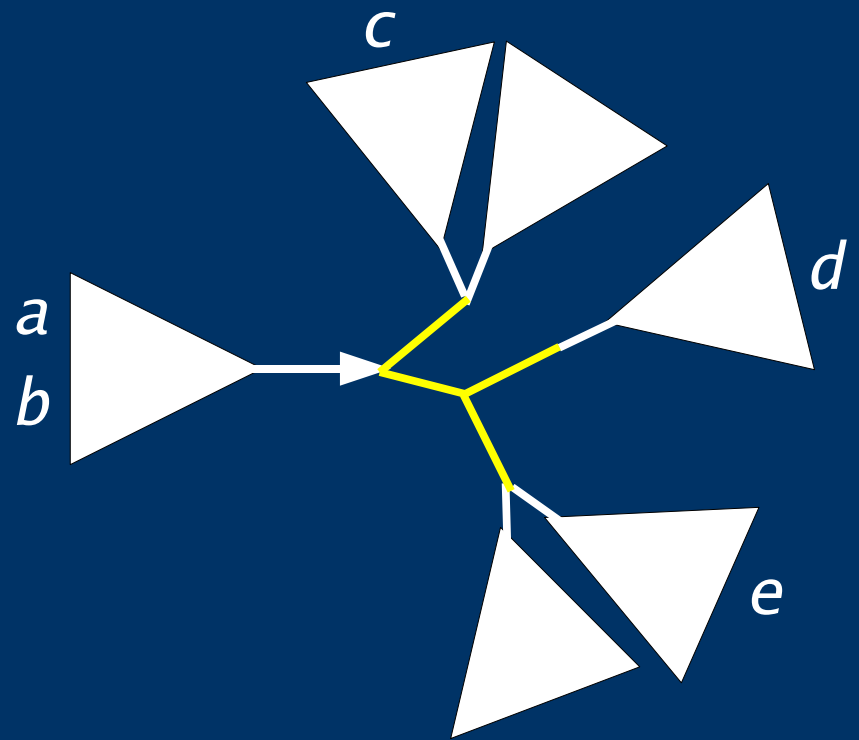
Expanding nodes



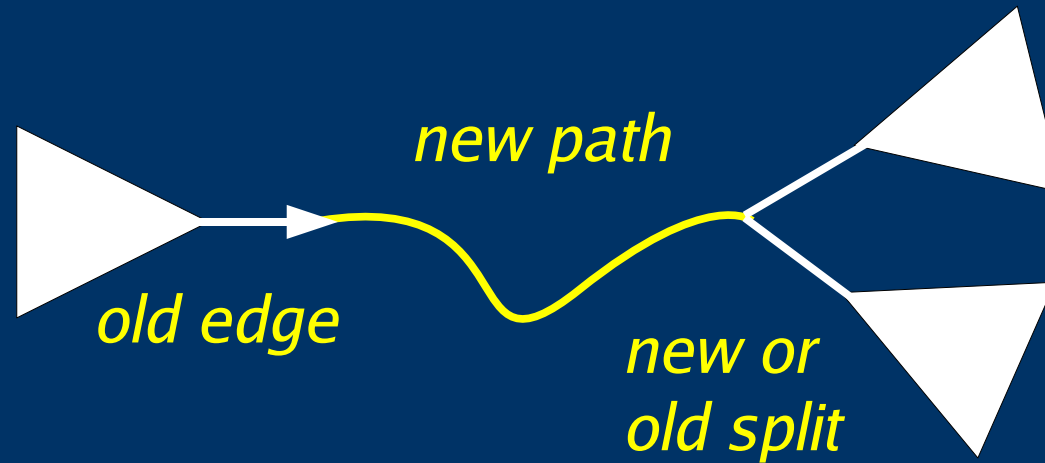
- Computable in time $O(n^2)$
- $O(d)$ new edges per original node
 - Still binary, so $O(n)$ nodes and edges
 - $O(n^2)$ pairs of claims

Too many butterflies

- The *new edges* introduce new butterflies
 - $ab|cd$ in original tree
 - $ac|de$ *not* in original tree



Extended claims



- Each *original butterfly* claimed by exactly two *extended claims*
 - Shared butterflies counted similar to regular claims
 - $O(n^2 d^2)$ pairs of extended claims
 - Can be enumerated in time $O(n^2 d^2)$

Conclusions

- Two algorithms for *quartet distance* for trees with *arbitrary degree*
- Complexity:
 - Time $O(n^3)$, space $O(n^2)$
 - Time $O(n^2 d^2)$, space $O(n^2)$
- Java implementation of both at:
<http://www.daimi.au.dk/~chrisc/qdist/>