DIAMANT autumn symposium 2022

Convex characters, algorithms and matchings

Steven Kelk

Department of Advanced Computing Sciences (DACS)
Maastricht University, Netherlands

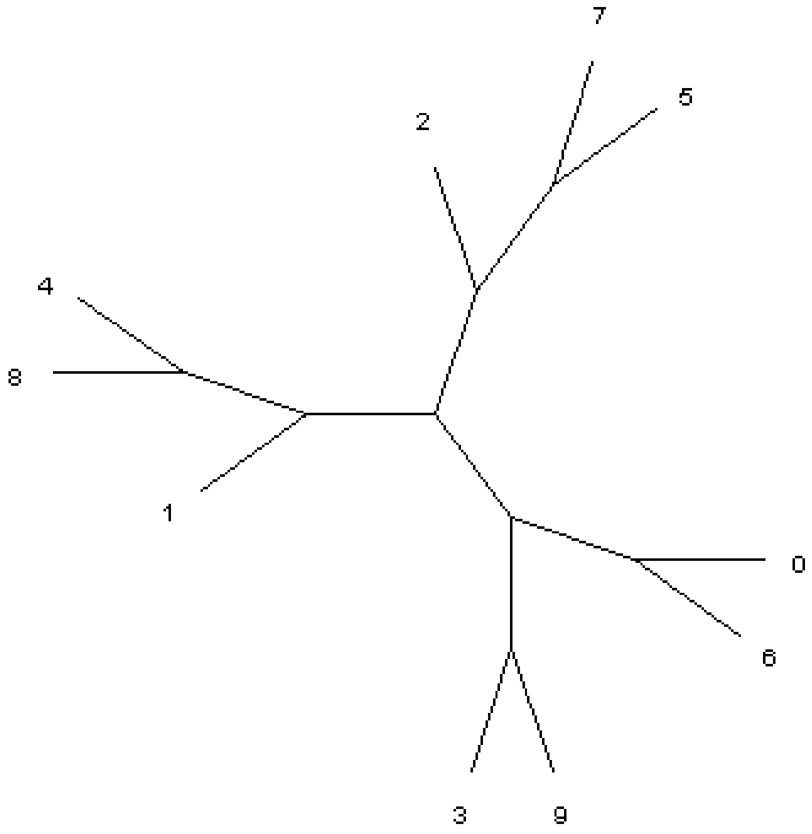
Based on joint work with Ruben Meuwese (DACS) and Stephan Wagner (Uppsala, Sweden)

https://arxiv.org/abs/2111.12632

Maastricht University

Dr. Andre Berger	algorithms and optimisation
Dr. Steve Chaplick	algorithmic graph theory
Dr. Christof Defryn	optimisation
Dr. Ioannis Diamantis	topology, geometry, knot theory
Dr. Barbara Franci	algorithmic game theory
Prof. dr. Alexander Grigoriev	combinatorial optimisation, algorithms and complexity
Prof. dr. Stan van Hoesel	combinatorial optimisation
Dr. Steven Kelk	algorithms and complexity
Dr. Stefan Maubach	algebraic geometry
Dr. Parag Mehta	algebra, topology
Dr. Matus Mihalak	algorithms, combinatorial optimisation
Prof. dr. Rudolf Müller	combinatorial optimisation, mechanism design
Dr. Marieke Musegaas	optimisation, cooperative game theory
Dr. Lars Rohwedder	algorithms and optimisation
Dr. Marc Schröder	game theory and optimisation
Dr. Georgios Stamoulis	algorithms, optimisation and complexity
Dr. Mathias Staudigl	optimisation, computational game theory
Prof. dr. Frank Thuijsman	game theory and operations research
Prof. dr. Dries Vermeulen	game theory
Prof. dr. Tjark Vredeveld	algorithms and optimisation
Dr. Tom van der Zanden	combinatorial optimisation, algorithms and complexity

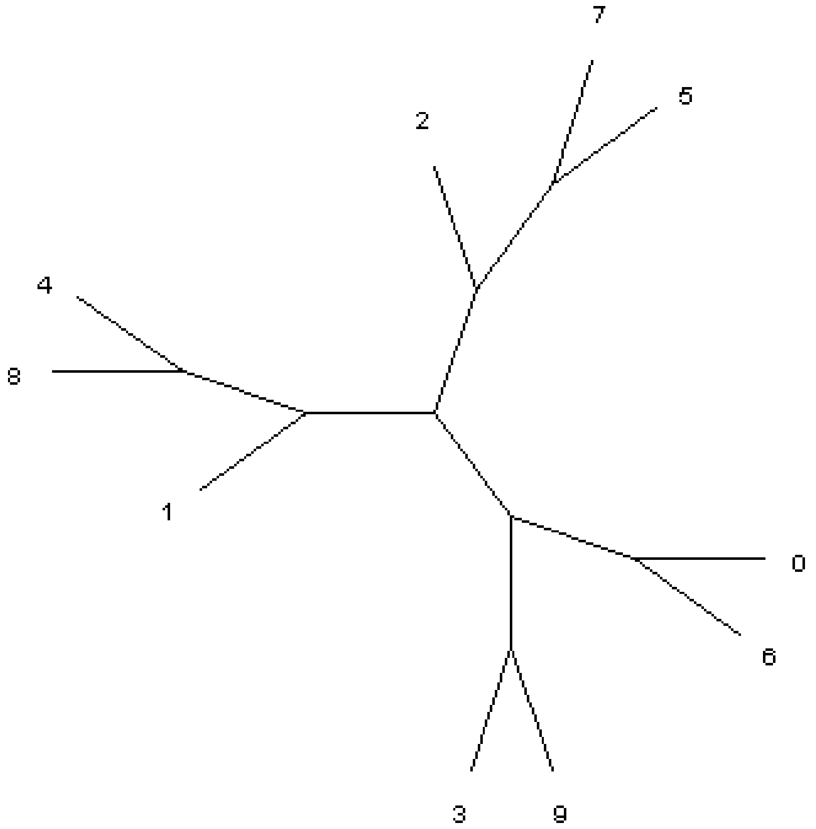
- Phylogenetic trees summarise the evolution of a set of species *X.*

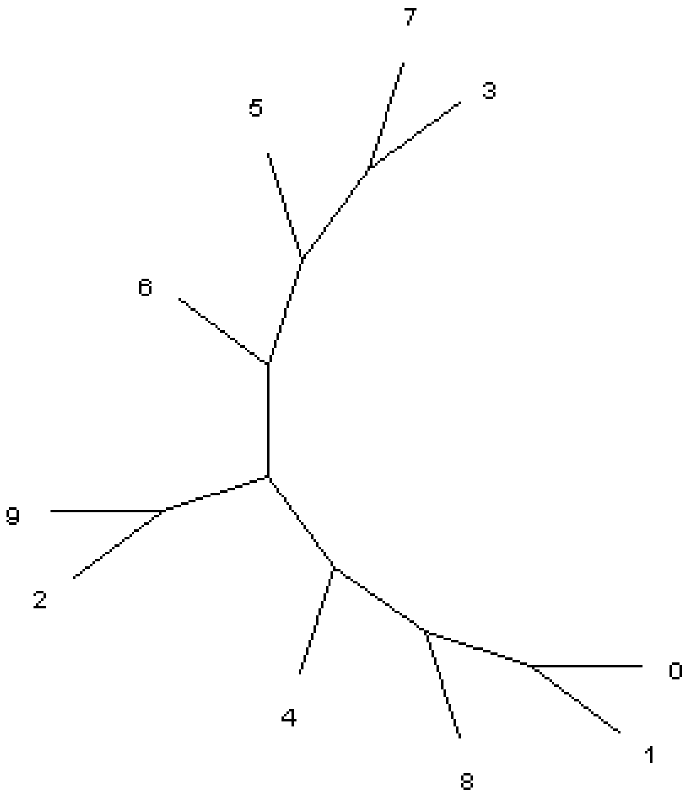


- The central goal of phylogenetics is to *infer* these trees from e.g. DNA data.

- However, phylogenetics software often generates several topologically distinct (*“incongruent”*) trees.

- Important to quantify incongruence i.e. in how far two (or more) trees differ from each other topologically.


7
5
2
4
8
1
0
6
3
9

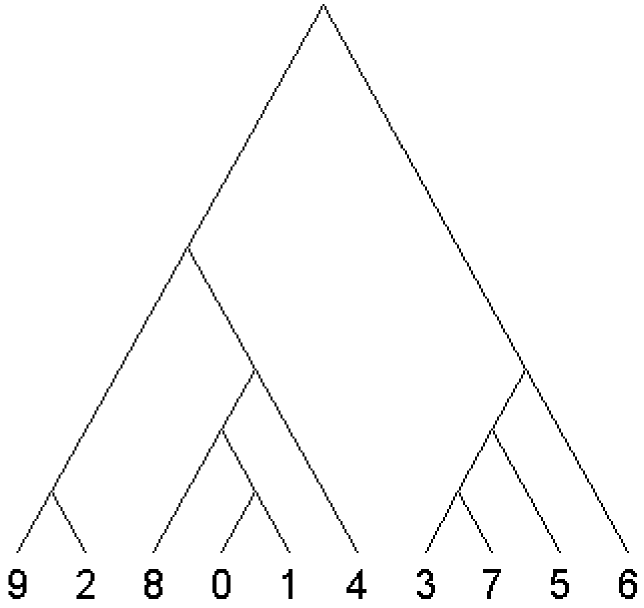

7
3
5
6
9
2
0
4
1
8

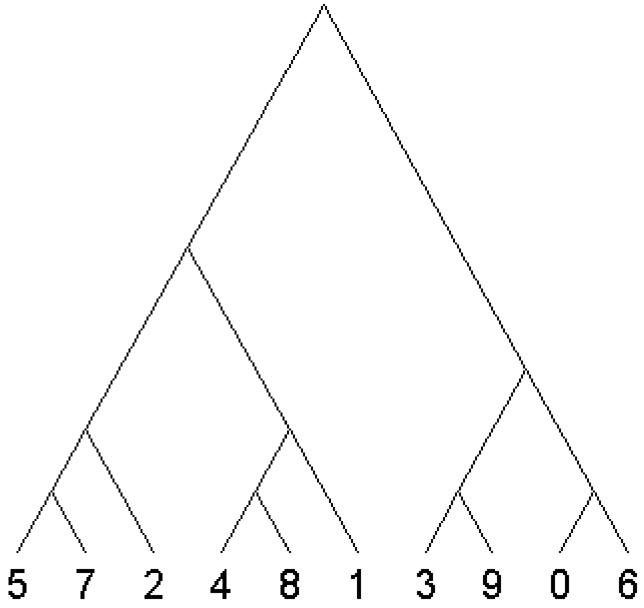
- ***Optimization problem:* “maximum parsimony distance on 2 colours”**

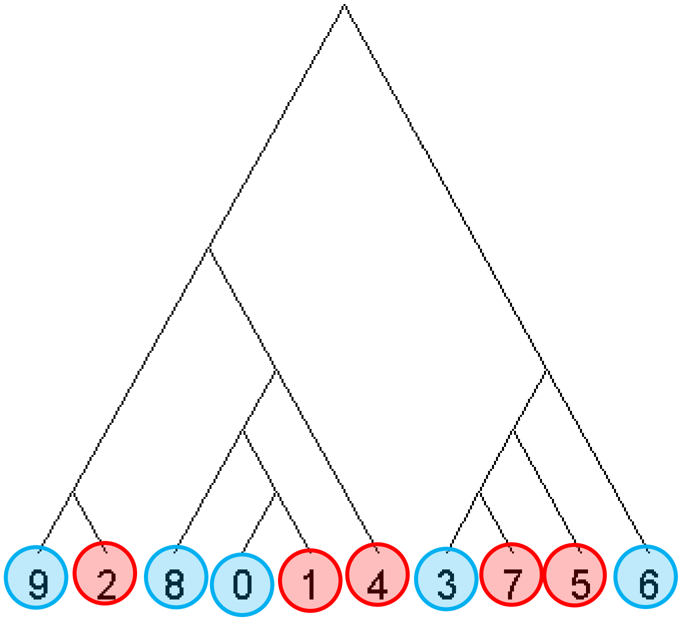
- Given two trees *T, T’* on leaf labels *X*, this asks us – informally! - to find a colouring of *X* with two colours {red, blue} such that in one of the trees the colouring induces ‘many’ bichromatic edges, and in the other tree the colouring induces ‘few’ bichromatic edges.

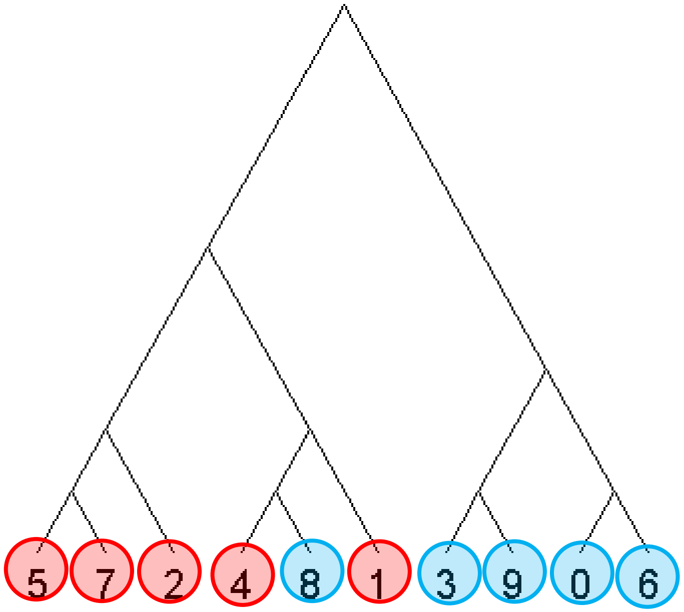
- The goal is to maximize the absolute difference in the number of induced bichromatic edges.

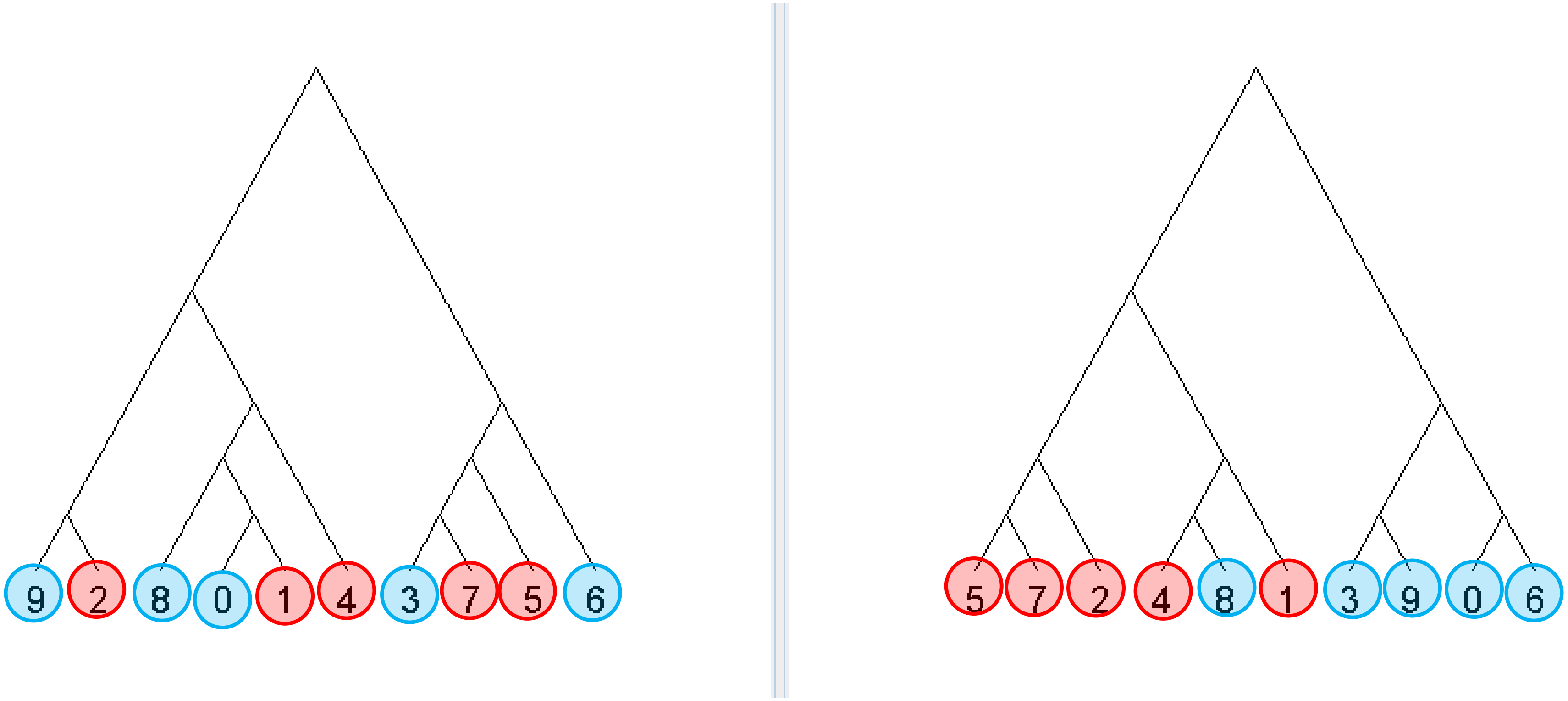
- Quick example, then formal definitions follow.


9 2 8 0 1 4 3 7 5 6

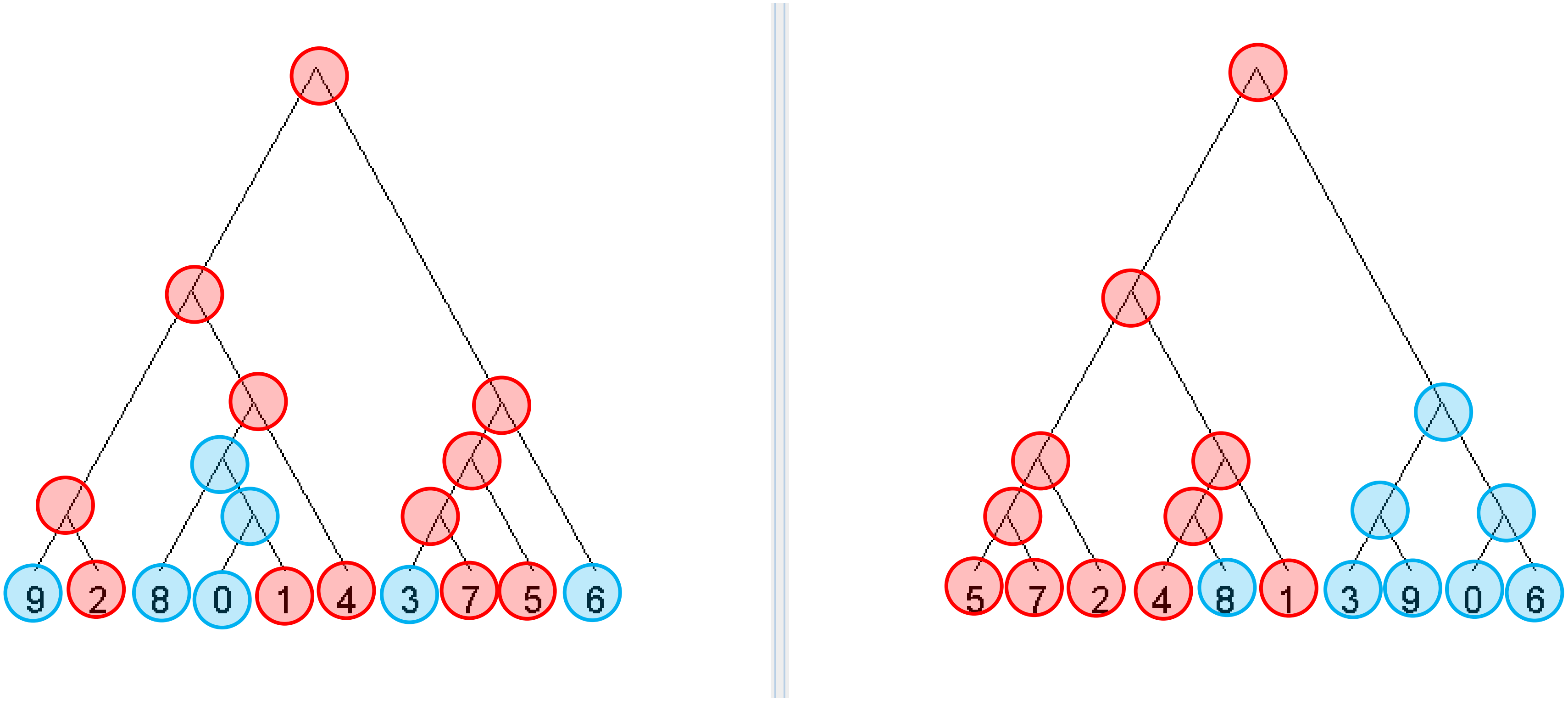

5 7 2 4 8 1 3 9 0 6


9
2
8
0
1
4
3
7
5
6

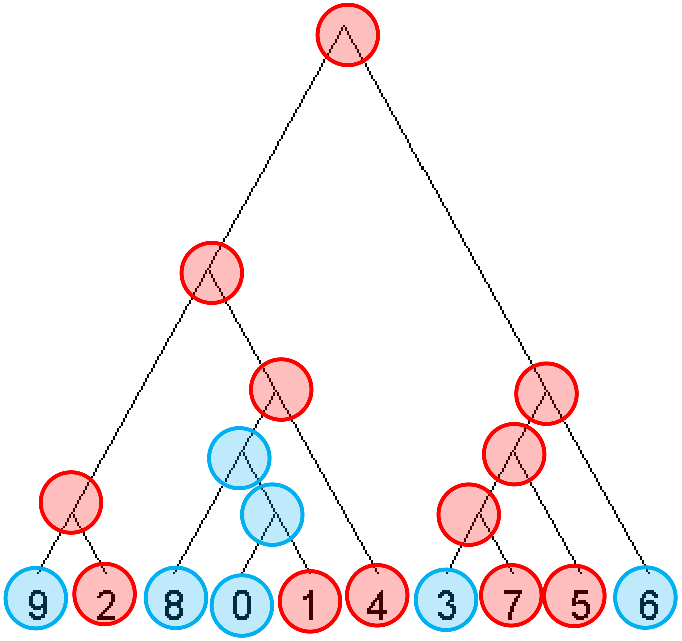

5
7
2
4
8
1
3
9
0
6



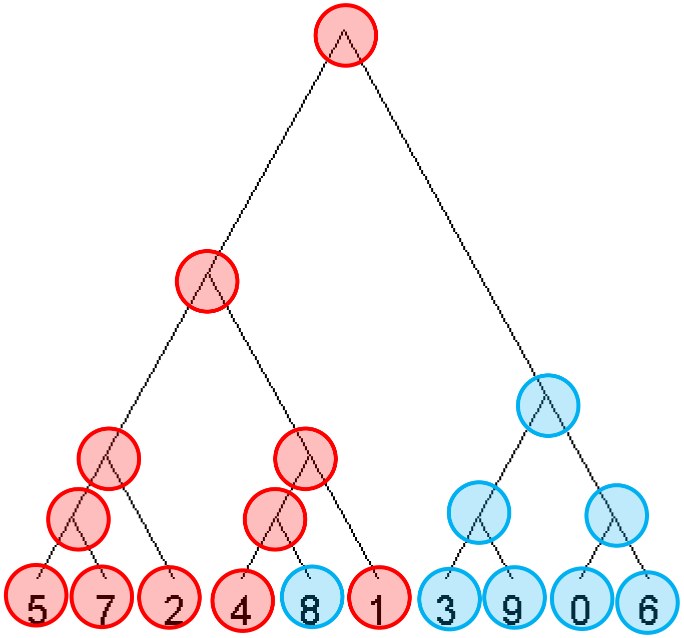
Each tree then colours its internal nodes
to minimize the number of bichromatic edges



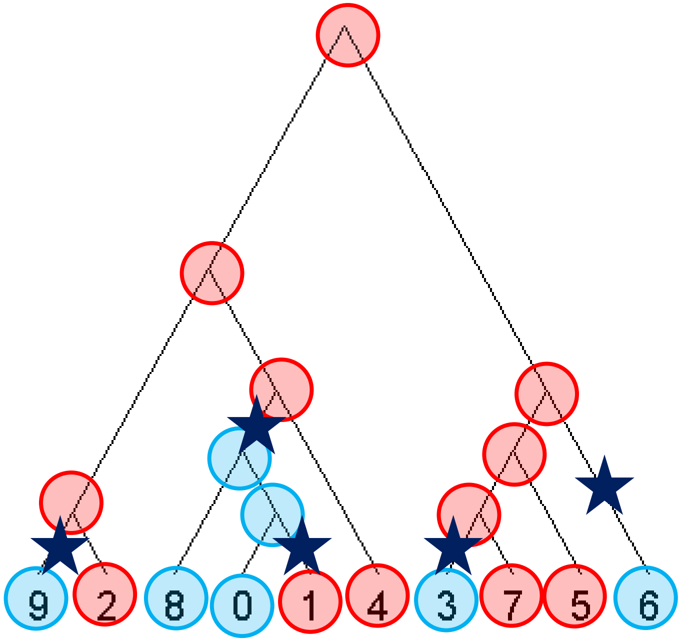
Each tree then colours its internal nodes
to minimize the number of bichromatic edges



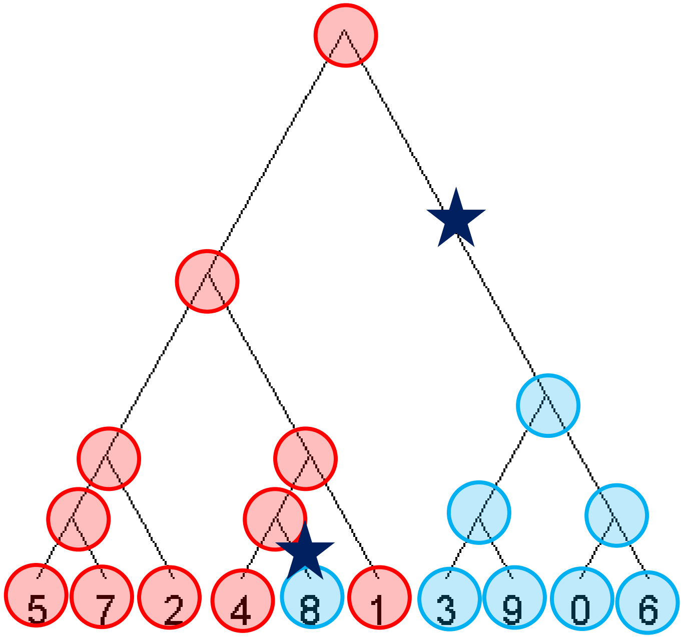
5 bichromatic edges



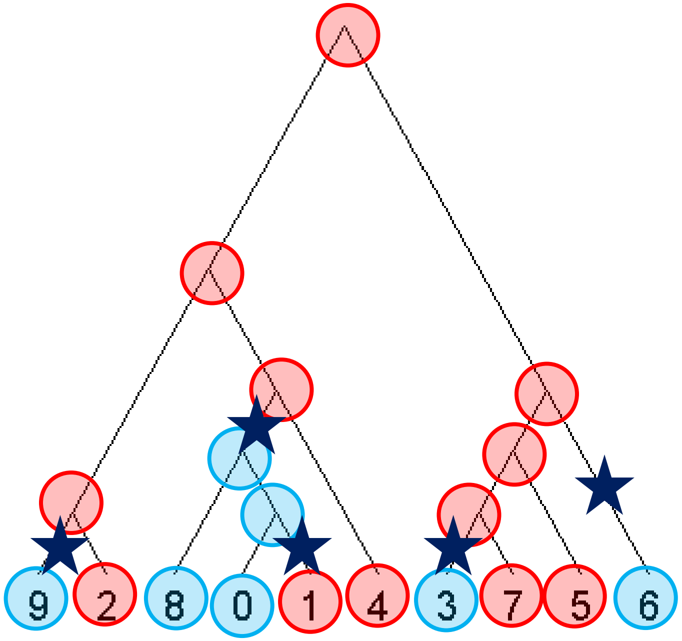
2 bichromatic edges



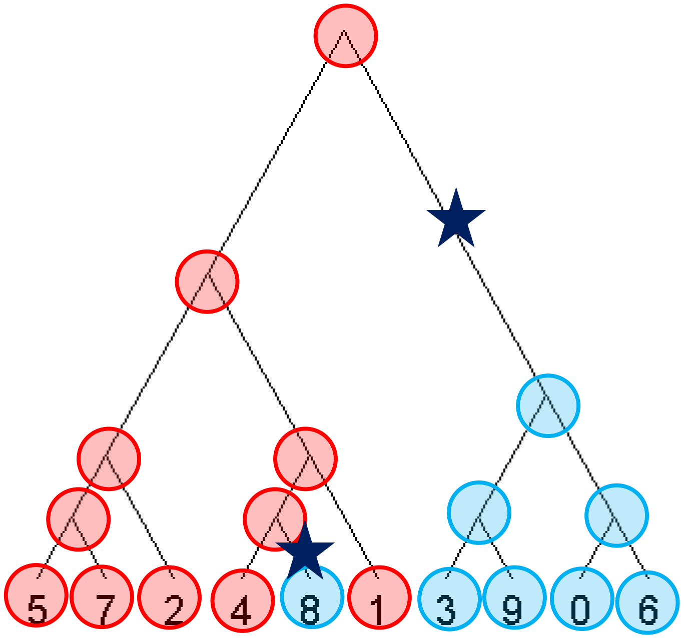
5 bichromatic edges



2 bichromatic edges

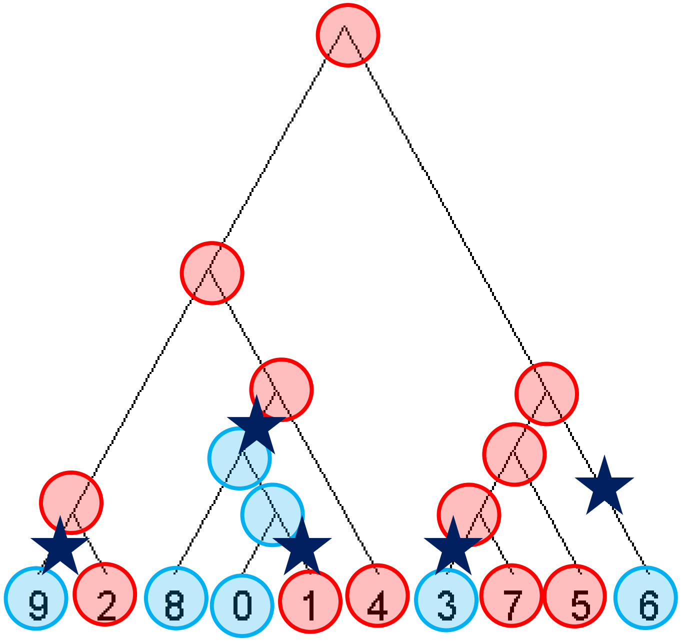


5 bichromatic edges

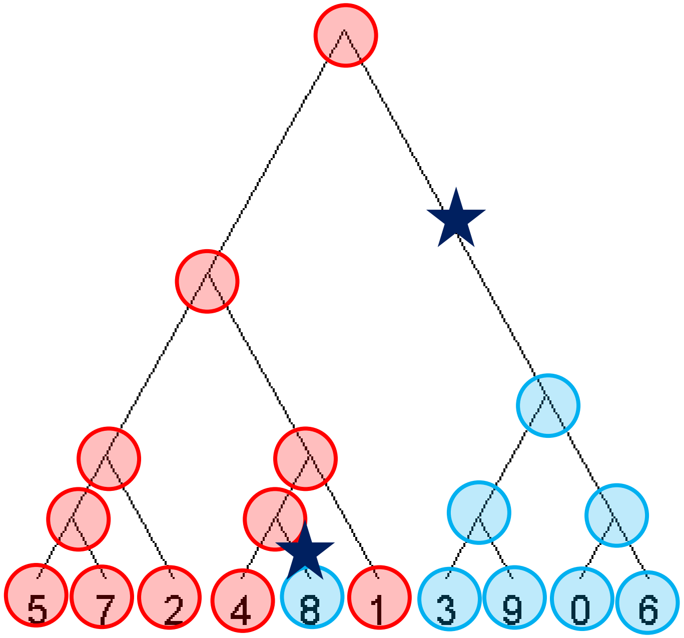


2 bichromatic edges

Absolute difference is 3



5 bichromatic edges



2 bichromatic edges

Absolute difference is 3

No leaf colouring can create a bigger absolute difference, so the trees have distance 3

- A *character f* is simply a colouring of *X*.
    - Can also be viewed simply as a partition of *X*, where colours = blocks.
    - I will henceforth refer to characters as *X-colourings.*

- An *extension* of an X-colouring *f* to a tree *T* (on *X*), is an expansion of *f* to also include the interior nodes of *T*.

- The *parsimony score* of a tree *T* (on *X*), with respect to *f*, is the minimum number of bichromatic edges, ranging over all extensions of *f* to *T*. This is denoted $\ell_f(T)$. An extension is *optimal* if it achieves this minimum.

- Given *T* and *T'*, both on *X*, we want to compute

$$dmp2(T,T') = Max_f \mid \ell_f(T) - \ell_f(T') \mid$$

….where here f ranges over all 2-colour *X*-colourings (i.e. bipartitions of X).

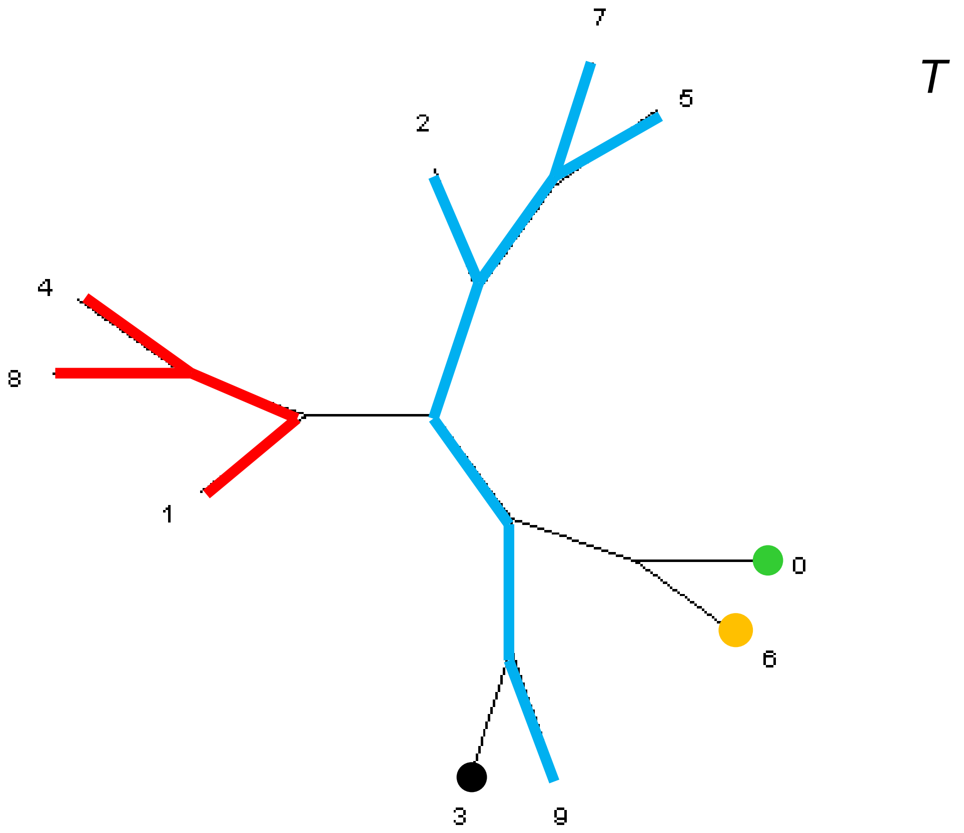
- NP-hard (and APX-hard) problem! Trivial $O^*(2^n)$ algorithm.

- Today we show an improvement to O*(1.6181^n), where here n=|X|.

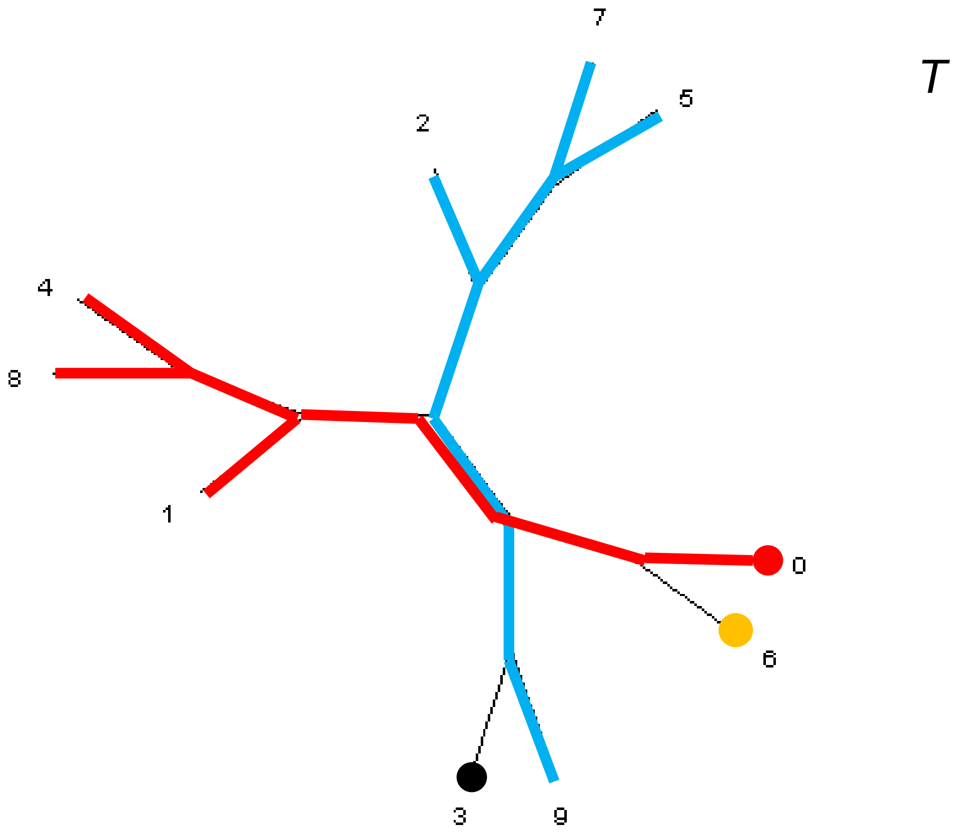
- … and then to O*(1.5895^n).

- These results are based on enumeration of so-called *convex X*-colourings ("convex characters" in the phylogenetics literature).

- An *X*-colouring is *convex* on *T*, if the spanning trees induced by the colours – one spanning tree per colour - are vertex disjoint in *T*.



Convex *X*-colouring { {1,8,4}, {3}, {9,2,7,5}, {0}, {6} } on *T*
- spanning trees are disjoint



Non-convex *X*-colouring { {1,8,4,0}, {3}, {9,2,7,5}, {6} } on *T*
- spanning trees are not disjoint

- Today we show an improvement to O*(1.6181^n), where here n=|X|.

- … and then to O*(1.5895^n).

- These results are based on enumeration of so-called *convex X*-colourings ("convex characters" in the phylogenetics literature).

- An X-colouring is convex on T, if the spanning trees induced by the colours – one spanning tree per colour - are vertex disjoint in T.

- Note that dmp2 seeks a 2-colour X-colouring that maximizes absolute difference in the number of induced bichromatic edges. These 2-colourings are not necessarily convex! We will enumerate convex X-colourings and then carefully project them back onto 2-colour X-colourings.

- This also gives us an interesting corollary!

- Recall: in a graph, a *matching* is simply a subset of mutually disjoint edges.

- Arbitrary trees on *n* nodes can have O(1.6181^n) matchings (consider: paths), this is well known.

- Regular 3-trees have O(1.5538^n) matchings, where the base of the exponent is $\sqrt{1+\sqrt{2}}$. This is also known.

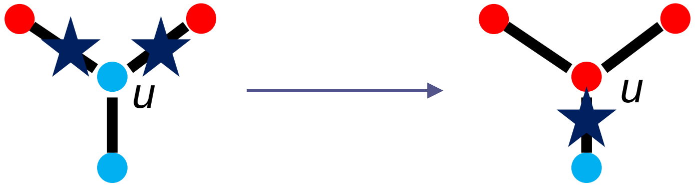
- New corollary: **trees with maximum degree 3, where there are no adjacent degree-2 nodes, have at most O(1.5895^n) matchings, and this bound is <u>sharp</u>.**

- But let’s first start at the beginning.

- Consider an *optimal* 2-colour *X*-colouring f_2, i.e. one which maximizes the absolute difference of parsimony scores between the input trees *T* and *T'*.

- Consider, now, an optimal extension of f_2 to *T* (i.e. an extension with a minimum number of bichromatic edges).

- Suppose there is an internal node *u* of T where two or three of its neighbours have a different colour to *u*…



- Then we get a contradiction on the assumed optimality of the extension; if you flip the colour of *u*, you get an extension with fewer bichromatic edges.

- As a result, we can assume the existence of an optimal 2-colour *X*-colouring, and an optimal extension of that *X*-colouring, in which internal nodes of *T* always lie on the interior of a red path or on the interior of a blue path.

- Next: observe that deleting the bichromatic edges in an optimal extension, on the tree with lower parsimony score, induces a new partition of *X*.

- Here, the induced partition is {{5,7,2,4,1}, {8}, {3,9,0,6}}.

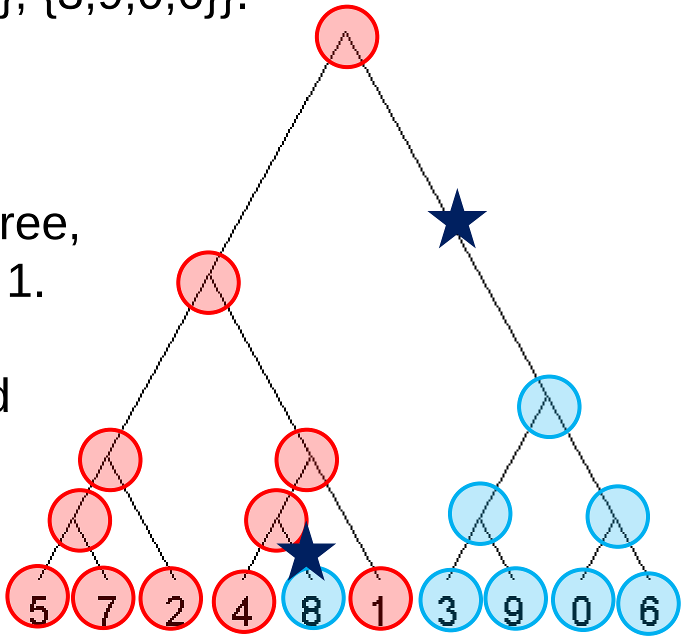
- What if we flip the colour of 8, to red?

- Parsimony score drops by *at least* 1 in this tree, but can decrease in the other tree by *at most* 1.

Recall: we assumed that this tree already had the lower parsimony score.

So the absolute difference between the trees does not decrease.

So we have a new optimal 2-colour *X*-colouring (that induces fewer singleton components)!



2 bichromatic edges

- Next: observe that deleting the bichromatic edges in an optimal extension, on the tree with lower parsimony score, induces a new partition of *X*.

- Here, the induced partition is {{5,7,2,4,1}, {8}, {3,9,0,6}}.

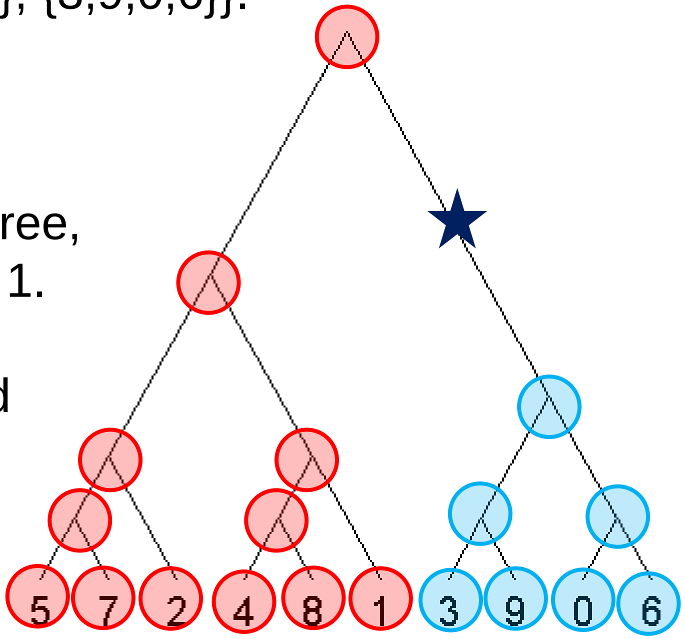
- What if we flip the colour of 8, to red?

- Parsimony score drops by *at least* 1 in this tree, but can decrease in the other tree by *at most* 1.

Recall: we assumed that this tree already had the lower parsimony score.

So the absolute difference between the trees does not decrease.

So we have a new optimal 2-colour *X*-colouring (that induces fewer singleton components)!

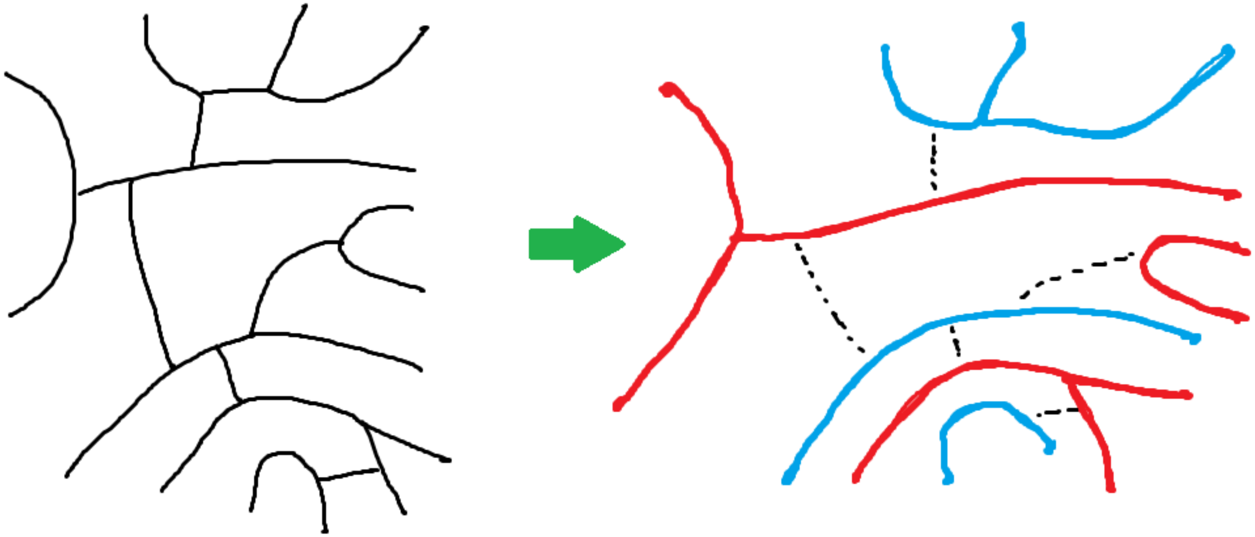


1 bichromatic edge

- By iterating this process, we eventually arrive at an optimal 2-colour *X*-colouring, and an optimal extension of that X-colouring, where…

    - Every internal node lies on the interior of a red path or a blue path;

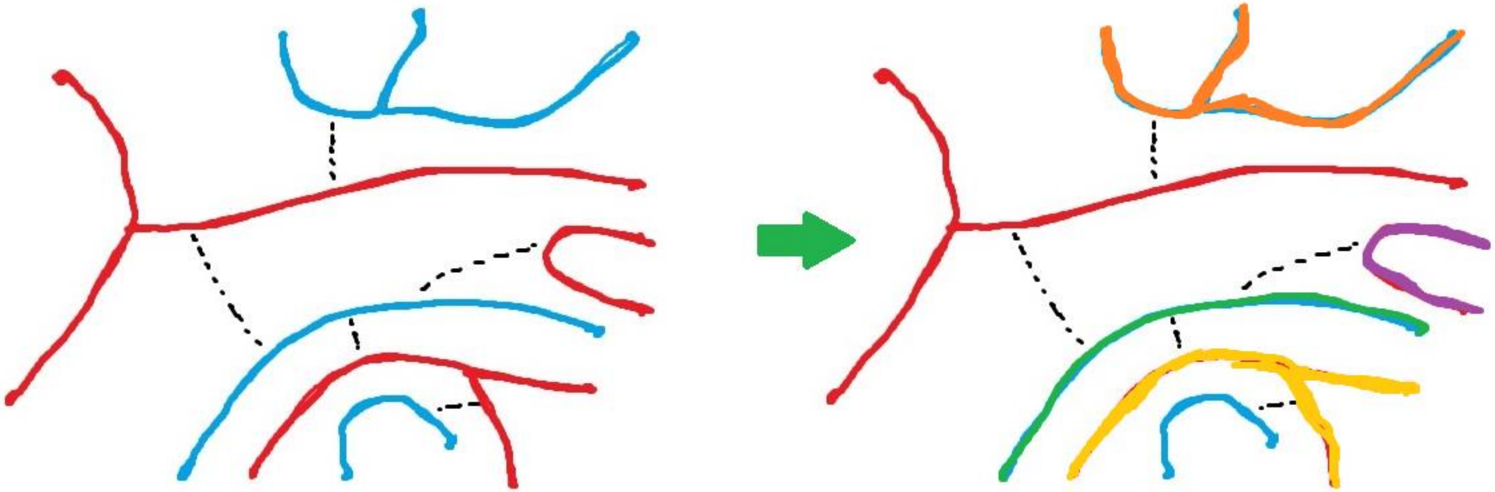
    - The new partition of *X* induced by deleting bichromatic edges of an optimal extension of the *X*-colouring, is such that every block of the partition contains at least 2 labels from *X*.



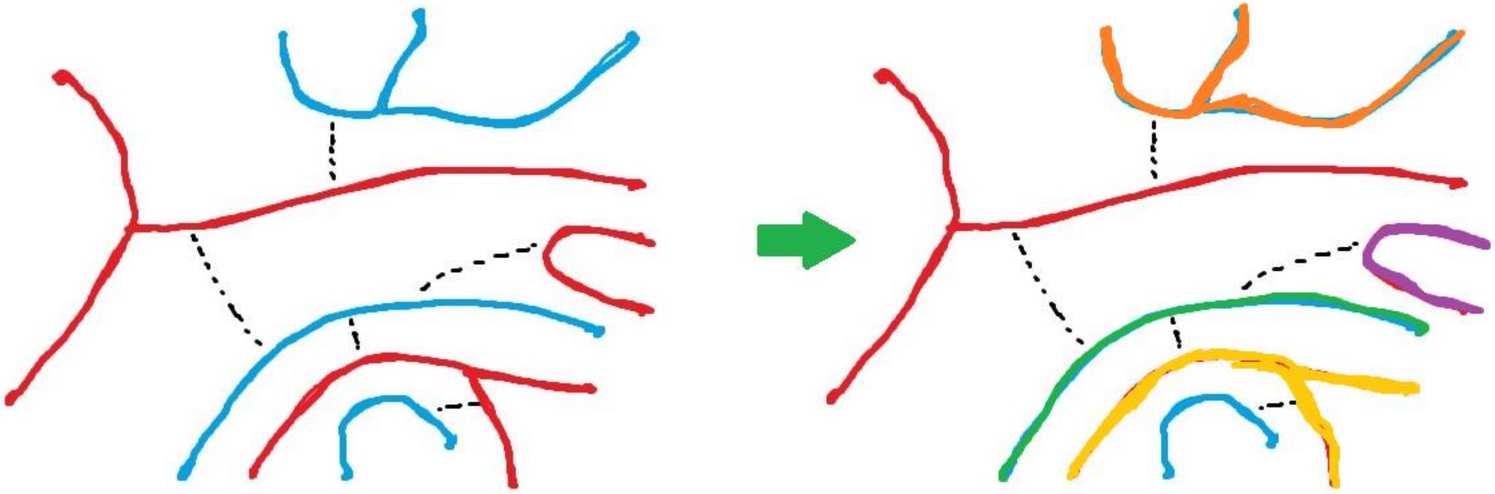
- We can relabel the blocks of the new partition of *X* induced by deleting bichromatic edges, by unique colours. So if there were *b* bichromatic edges, there are *b+1* colours.

- Such an X-colouring is a convex *X*-colouring, in which every colour appears on at least 2 labels of *X*, and such that the spanning trees for these colours cover every internal node of *T*.

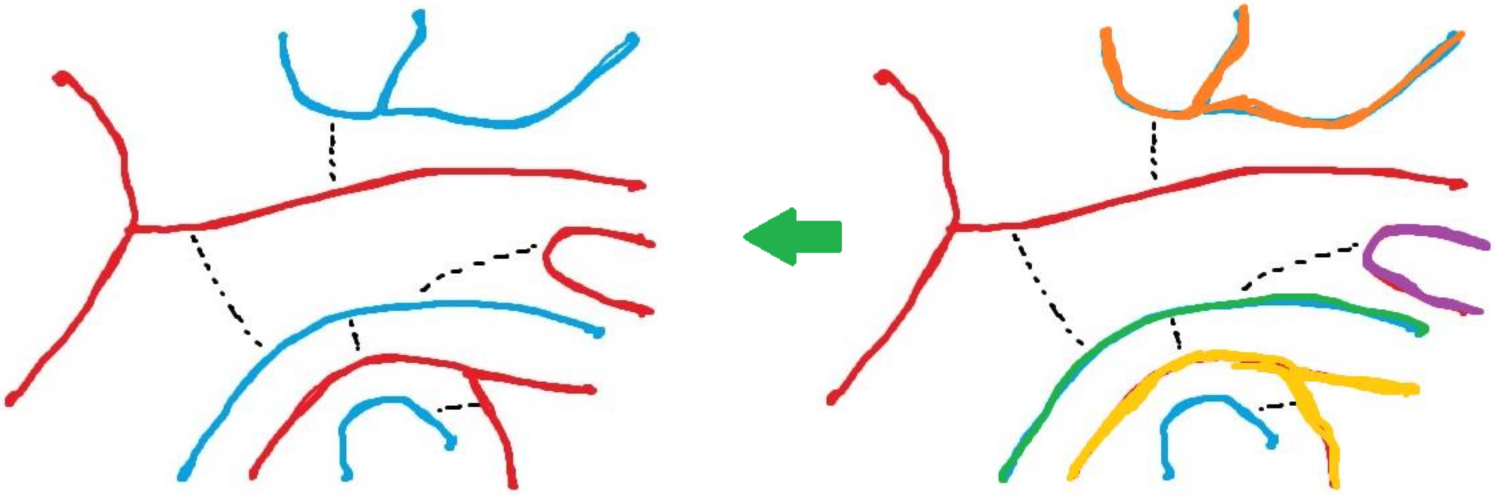
- (In fact, the spanning trees for these colours are provably the only optimal extension for this convex *X*-colouring.)



- Note that, if you could find this specific convex *X*-colouring, you could easily map it back without ambiguity to the optimal 2-colour *X*-colouring (this is critical!)



- Note that, if you could find this specific convex *X*-colouring, you could easily map it back without ambiguity to the optimal 2-colour *X*-colouring (this is critical!)



- There are Ɵ(1.6181^n) convex *X*-colourings with at least two labels from *X* per colour, and they can be listed efficiently [K. and Stamoulis, 2019]

- This gives us the simple, enumeration-based algorithm we need with running time Ɵ*(1.6181^n).

1) 'Guess' the tree from {T,T'} with lower parsimony score at optimality;

2) Loop through all convex X-colourings (on that tree) with at least two labels from X per colour:

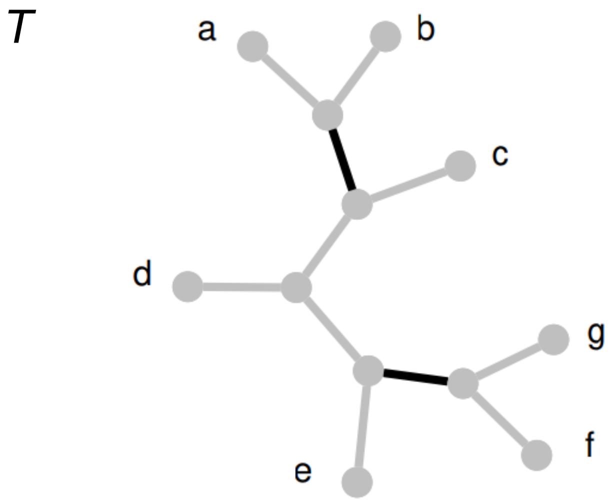
- *…In each case, map it to the corresponding, uniquely defined 2-colour X-colouring, and note how good this 2-colour X-colouring is in terms of the absolute difference in parsimony scores between the two trees.*

3) Pick the best such 2-colour X-colouring that we find.

- **Let’s do better!**

- We will still enumerate convex *X*-colourings (with each colour appearing on at least two labels of *X*), but we will discard some ‘useless’ part of this space.

- First: we can prove that if you take an optimal extension of an optimal convex *X*-colouring (i.e. one that maps back to an optimal 2-colour *X*-colouring), the bichromatic edges are a *matching* on *T*. Also, no matching edge is incident to a leaf of *T*.



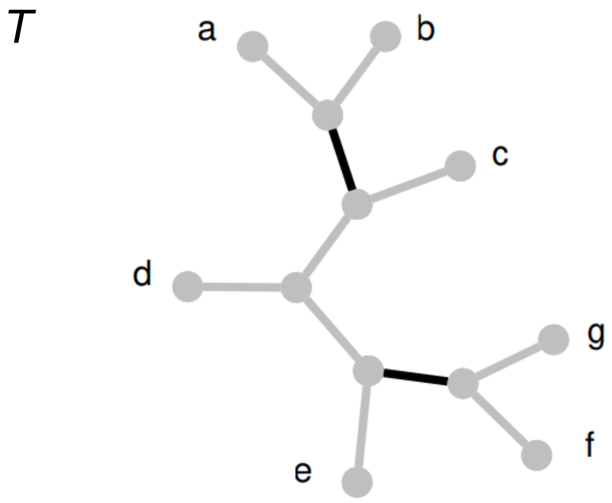
convex X-colouring {{a,b}, {c,d,e}, {f,g}}

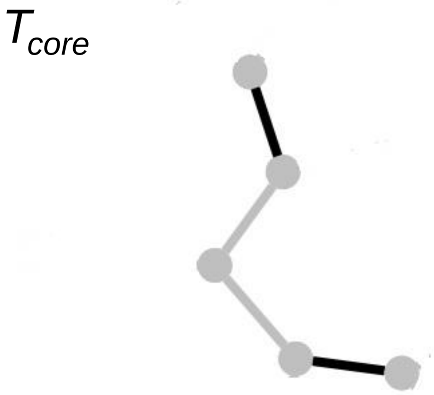
bichromatic edges are shown in black, forming a matching

- Let T_{core} be the tree obtained by deleting the leaves of T.

- The convex X-colourings we are (potentially) interested in, are in bijection with the space of matchings on T_{core}.

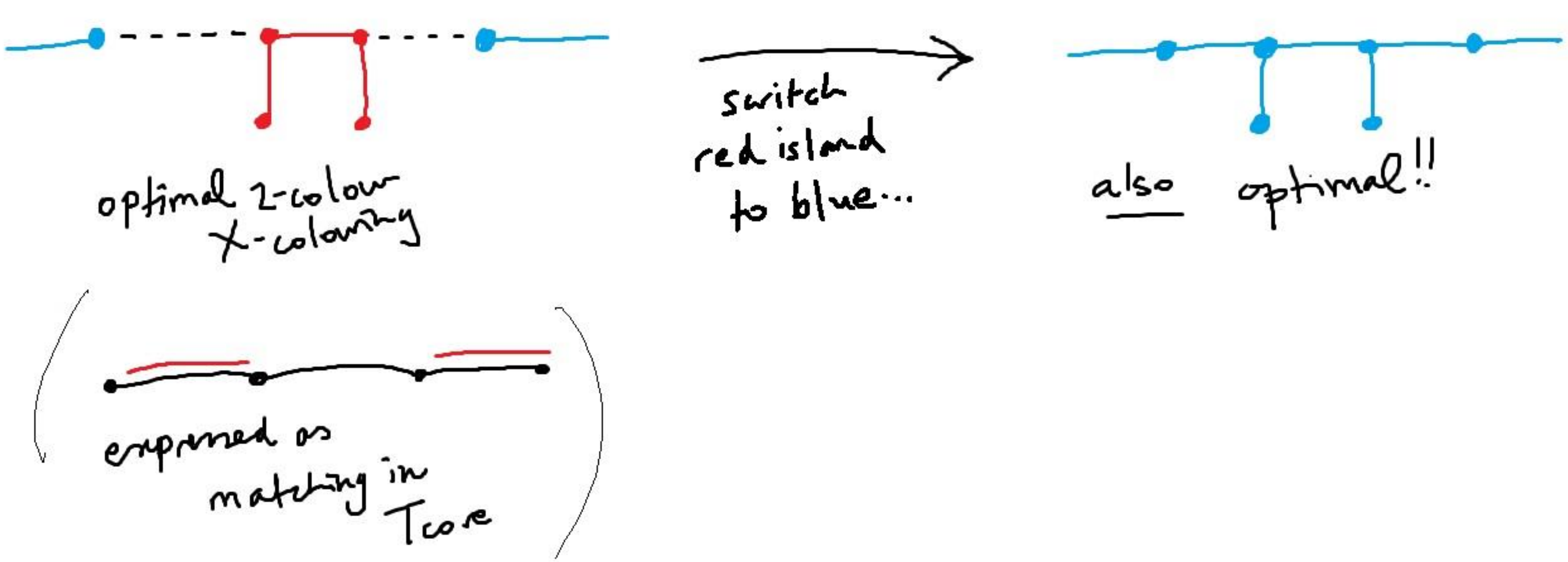
- So upper bounds on the number of matchings in T_{core}, can be translated into bounds on the number of relevant convex X-colourings in T, and thus to new bounds on the running time of the dmp2 algorithm.





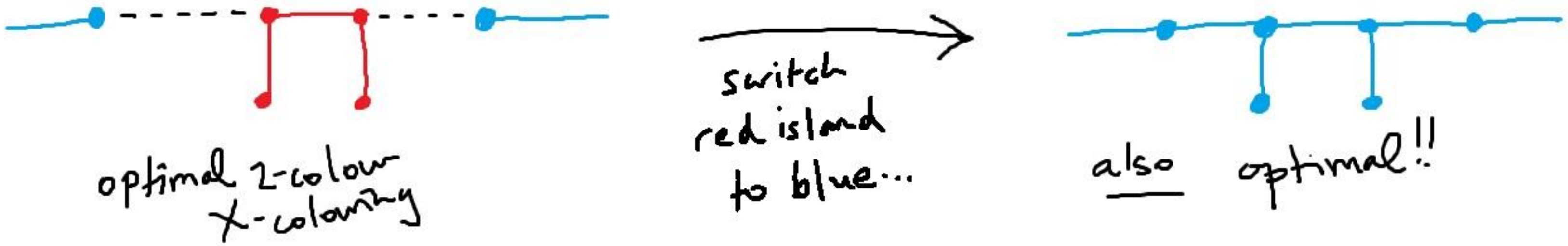
- Problem: if T_{core} is a path, then there can still be $\Theta(1.6181^n)$ matchings and thus, also, an equal number of relevant convex X-colourings on T; does not help to improve the bound ☹

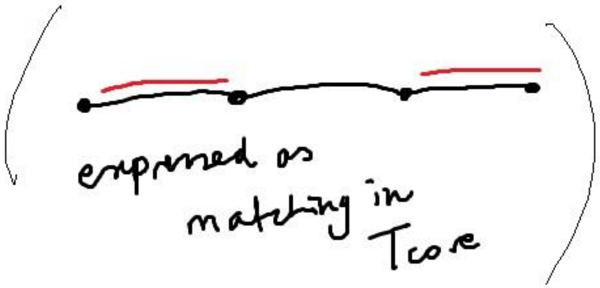
- But! We can leverage some additional insights about the structure of optimal solutions to the dmp2 problem.


optimal 2-colour X-colouring
switch red island to blue...
also optimal!!
expressed as matching in Tcore

- Problem: if T_{core} is a path, then there can still be Ө(1.6181^n) matchings and thus, also, an equal number of relevant convex *X*-colourings on T; does not help to improve the bound ☹

- But! We can leverage some additional insights about the structure of optimal solutions to the dmp2 problem.



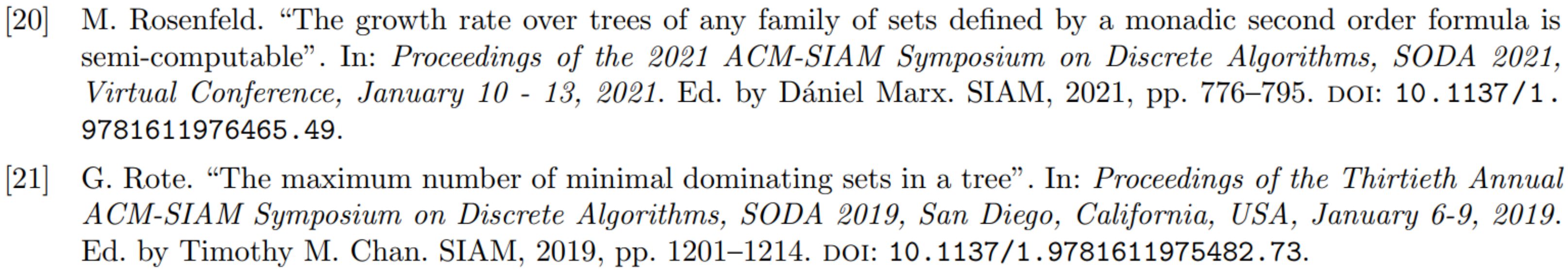


Conclusion: it is not necessary to consider any matchings in T_{core} that have this structure, so exclude them!

- Let us call matchings that do not have this ‘island’ sub-structure, *legal matchings.*

- How many legal matchings can there be in a tree with *n* nodes?

- We can establish a recurrence for this, and subsequently bound the rate of growth of the recurrence using techniques from these SODA articles:



[20] M. Rosenfeld. “The growth rate over trees of any family of sets defined by a monadic second order formula is semi-computable”. In: *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms, SODA 2021, Virtual Conference, January 10 - 13, 2021*. Ed. by Dániel Marx. SIAM, 2021, pp. 776–795. DOI: `10.1137/1.9781611976465.49`.

[21] G. Rote. “The maximum number of minimal dominating sets in a tree”. In: *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2019, San Diego, California, USA, January 6-9, 2019*. Ed. by Timothy M. Chan. SIAM, 2019, pp. 1201–1214. DOI: `10.1137/1.9781611975482.73`.

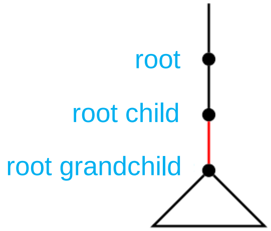


Figure 5: Legal matchings stored in b_0. A red edge represents an edge in the matching. As in the other figures in this section the edge entering from above represents the edge that enters the subtree in the original tree.

b_0 - root has one child, and one grandchild; edge between child and grandchild is a matching edge

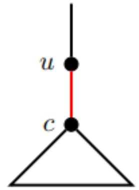


Figure 6: Legal matchings stored in b_1. A red edge represents an edge in the matching.

b_1 - root has one child, and the root-child edge is a matching edge

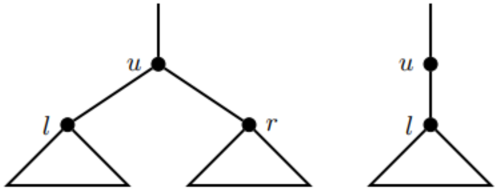


Figure 7: Legal matchings stored in a_0.

a_0 - legal matchings (excluding b_0) where there is no matching edge incident to the root

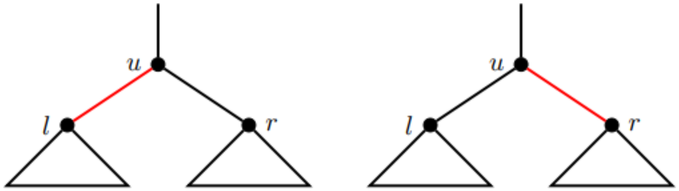
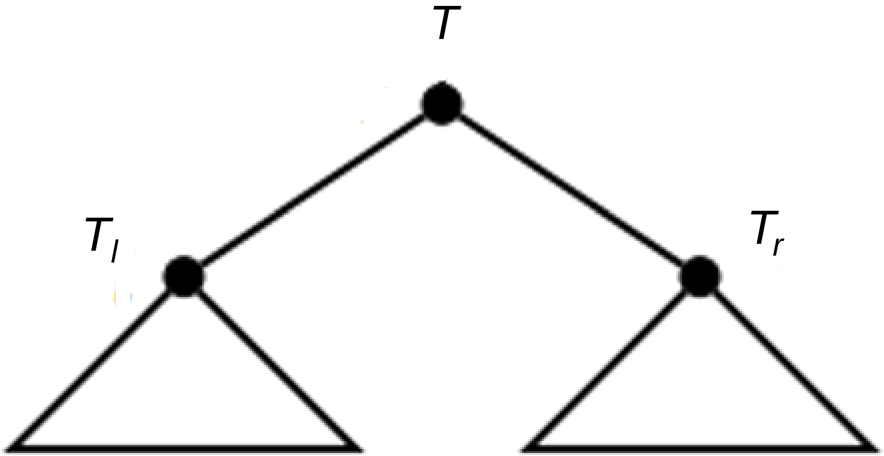


Figure 8: Legal matchings stored in a_1. A red edge represents an edge in the matching.

a_1 - legal matchings (excluding b_1) where there is a matching edge incident to the root

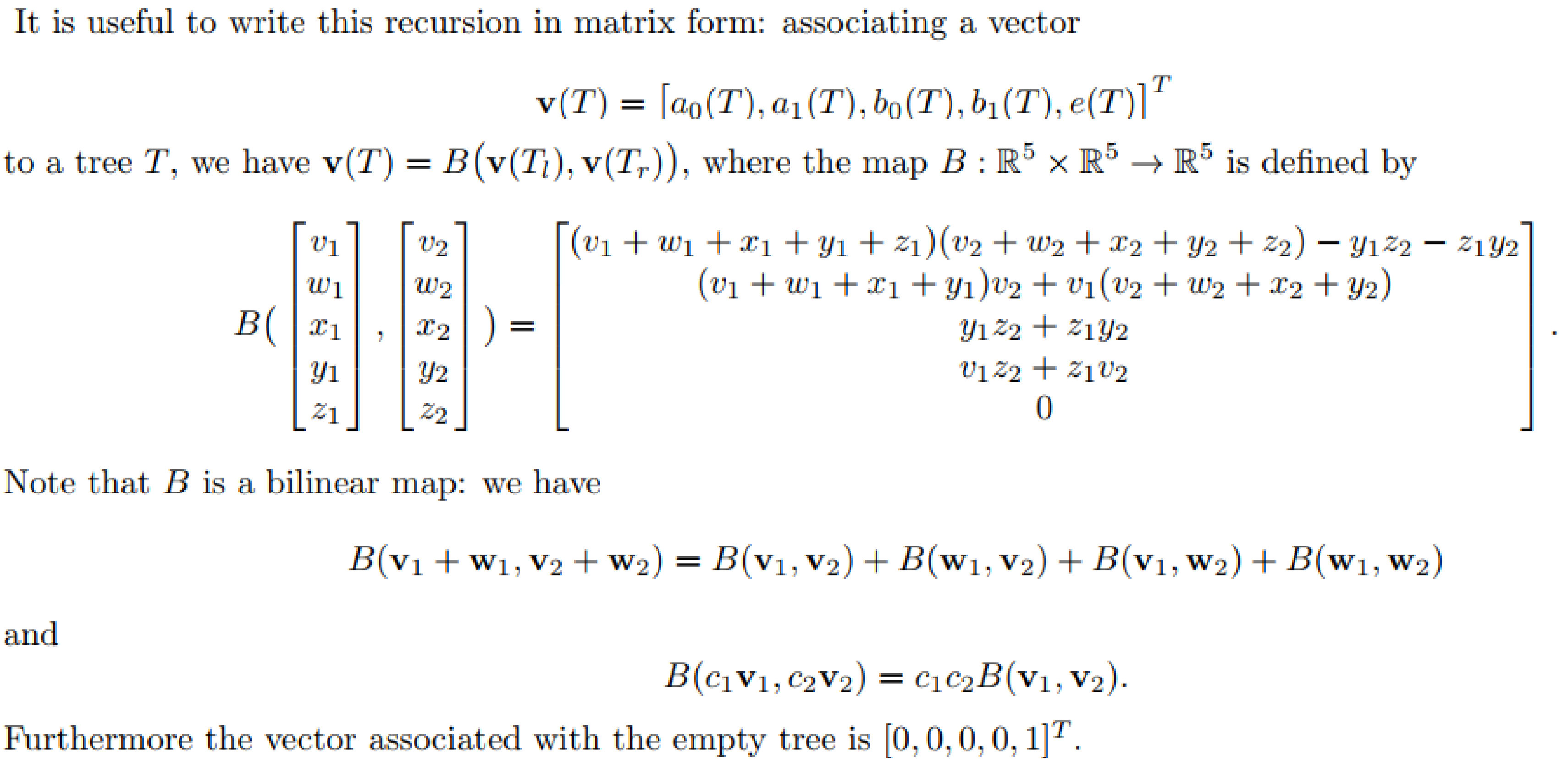
$$\begin{aligned} a_0(T) &= \big(a_0(T_l)+a_1(T_l)+b_0(T_l)+b_1(T_l)+e(T_l)\big)\big(a_0(T_r)+a_1(T_r)+b_0(T_r)+b_1(T_r)+e(T_r)\big) \\ &\qquad - b_1(T_l)e(T_r) - e(T_l)b_1(T_r), \\ a_1(T) &= a_0(T_l)\big(a_0(T_r)+a_1(T_r)+b_0(T_r)+b_1(T_r)\big) + \big(a_0(T_l)+a_1(T_l)+b_0(T_l)+b_1(T_l)\big)a_0(T_r), \\ b_0(T) &= b_1(T_l)e(T_r) + e(T_l)b_1(T_r), \\ b_1(T) &= a_0(T_l)e(T_r) + e(T_l)a_0(T_r). \end{aligned}$$

(Note: *e(T)* is 1 if *T* is the empty tree, and 0 otherwise.)



$$\begin{aligned}
a_0(T) &= \big(a_0(T_l)+a_1(T_l)+b_0(T_l)+b_1(T_l)+e(T_l)\big)\big(a_0(T_r)+a_1(T_r)+b_0(T_r)+b_1(T_r)+e(T_r)\big)\\
&\qquad - b_1(T_l)e(T_r) - e(T_l)b_1(T_r),\\
a_1(T) &= a_0(T_l)\big(a_0(T_r)+a_1(T_r)+b_0(T_r)+b_1(T_r)\big) + \big(a_0(T_l)+a_1(T_l)+b_0(T_l)+b_1(T_l)\big)a_0(T_r),\\
b_0(T) &= b_1(T_l)e(T_r) + e(T_l)b_1(T_r),\\
b_1(T) &= a_0(T_l)e(T_r) + e(T_l)a_0(T_r).
\end{aligned}$$

(Note: *e(T)* is 1 if *T* is the empty tree, and 0 otherwise.)



It is useful to write this recursion in matrix form: associating a vector

$$\mathbf{v}(T) = \left[a_0(T), a_1(T), b_0(T), b_1(T), e(T)\right]^T$$

to a tree T, we have $\mathbf{v}(T) = B\big(\mathbf{v}(T_l), \mathbf{v}(T_r)\big)$, where the map $B : \mathbb{R}^5 \times \mathbb{R}^5 \to \mathbb{R}^5$ is defined by

$$B\left(\begin{bmatrix} v_1 \\ w_1 \\ x_1 \\ y_1 \\ z_1 \end{bmatrix}, \begin{bmatrix} v_2 \\ w_2 \\ x_2 \\ y_2 \\ z_2 \end{bmatrix}\right) = \begin{bmatrix} (v_1+w_1+x_1+y_1+z_1)(v_2+w_2+x_2+y_2+z_2) - y_1z_2 - z_1y_2 \\ (v_1+w_1+x_1+y_1)v_2 + v_1(v_2+w_2+x_2+y_2) \\ y_1z_2 + z_1y_2 \\ v_1z_2 + z_1v_2 \\ 0 \end{bmatrix}.$$

Note that B is a bilinear map: we have

$$B(\mathbf{v}_1+\mathbf{w}_1, \mathbf{v}_2+\mathbf{w}_2) = B(\mathbf{v}_1,\mathbf{v}_2) + B(\mathbf{w}_1,\mathbf{v}_2) + B(\mathbf{v}_1,\mathbf{w}_2) + B(\mathbf{w}_1,\mathbf{w}_2)$$

and

$$B(c_1\mathbf{v}_1, c_2\mathbf{v}_2) = c_1c_2B(\mathbf{v}_1,\mathbf{v}_2).$$

Furthermore the vector associated with the empty tree is $[0,0,0,0,1]^T$.

Theorem 6. *The maximum number M_n of legal matchings in a tree with n nodes is $O(\alpha^n)$ with $\alpha = (13384 + 8\sqrt{2793745})^{1/22} \approx 1.58945$.*

Proof. There exists a set $\mathcal{S}$ of 62 5-dimensional vectors with nonnegative entries that have the following property:

(1) It contains the vector $[0, 0, 0, 0, 1/\alpha]^T$,

(2) For any pair of two vectors $\mathbf{v}_1, \mathbf{v}_2 \in \mathcal{S}$, the vector $B(\mathbf{v}_1, \mathbf{v}_2)$ lies in the set

$$conv_{\leq}(\mathcal{S}) = \left\{\mathbf{w} \in \mathbb{R}^5 \,:\, \mathbf{w} \geq \mathbf{0}, \mathbf{w} \leq \sum_{\mathbf{v}\in\mathcal{S}} c_{\mathbf{v}}\mathbf{v} \text{ for some constants } c_{\mathbf{v}} \geq 0, \sum_{\mathbf{v}\in\mathcal{S}} c_{\mathbf{v}} = 1\right\}.$$

Here, the inequalities hold componentwise. Note that $conv_{\leq}(\mathcal{S})$ is a bounded and convex set by construction.

Theorem 6. *The maximum number M_n of legal matchings in a tree with n nodes is $O(\alpha^n)$ with $\alpha = (13384 + 8\sqrt{2793745})^{1/22} \approx 1.58945$.*

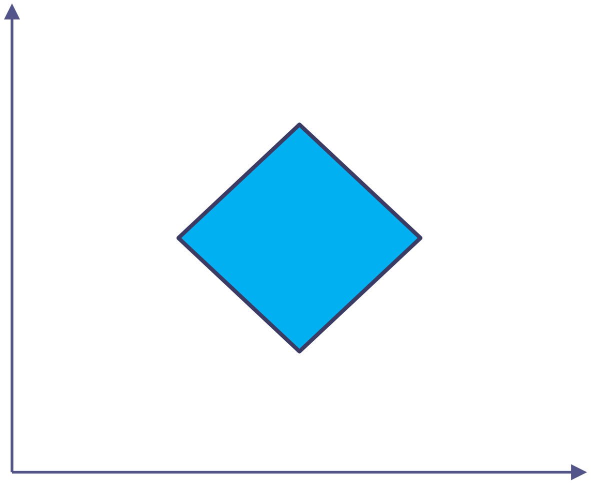
Proof. There exists a set $\mathcal{S}$ of 62 5-dimensional vectors with nonnegative entries that have the following property:

(1) It contains the vector $[0, 0, 0, 0, 1/\alpha]^T$,

(2) For any pair of two vectors $\mathbf{v}_1, \mathbf{v}_2 \in \mathcal{S}$, the vector $B(\mathbf{v}_1, \mathbf{v}_2)$ lies in the set

$$conv_{\le}(\mathcal{S}) = \left\{\mathbf{w} \in \mathbb{R}^5 : \mathbf{w} \ge \mathbf{0}, \mathbf{w} \le \sum_{\mathbf{v} \in \mathcal{S}} c_{\mathbf{v}}\mathbf{v} \text{ for some constants } c_{\mathbf{v}} \ge 0, \sum_{\mathbf{v} \in \mathcal{S}} c_{\mathbf{v}} = 1\right\}.$$

Here, the inequalities hold componentwise. Note that $conv_{\le}(\mathcal{S})$ is a bounded and convex set by construction.



Theorem 6. *The maximum number M_n of legal matchings in a tree with n nodes is $O(\alpha^n)$ with* $\alpha = (13384 + 8\sqrt{2793745})^{1/22} \approx 1.58945$.

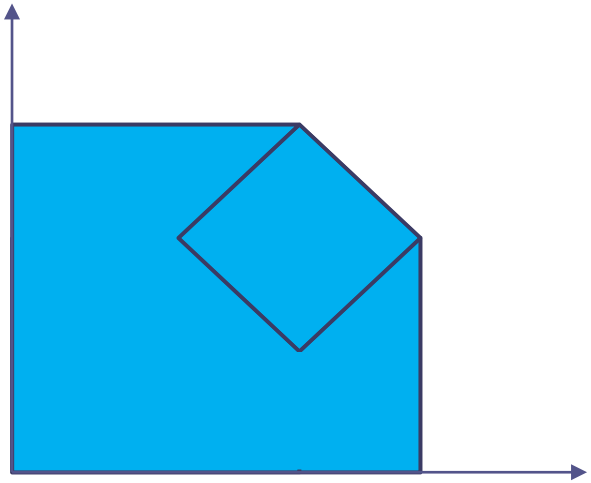
Proof. There exists a set $\mathcal{S}$ of 62 5-dimensional vectors with nonnegative entries that have the following property:

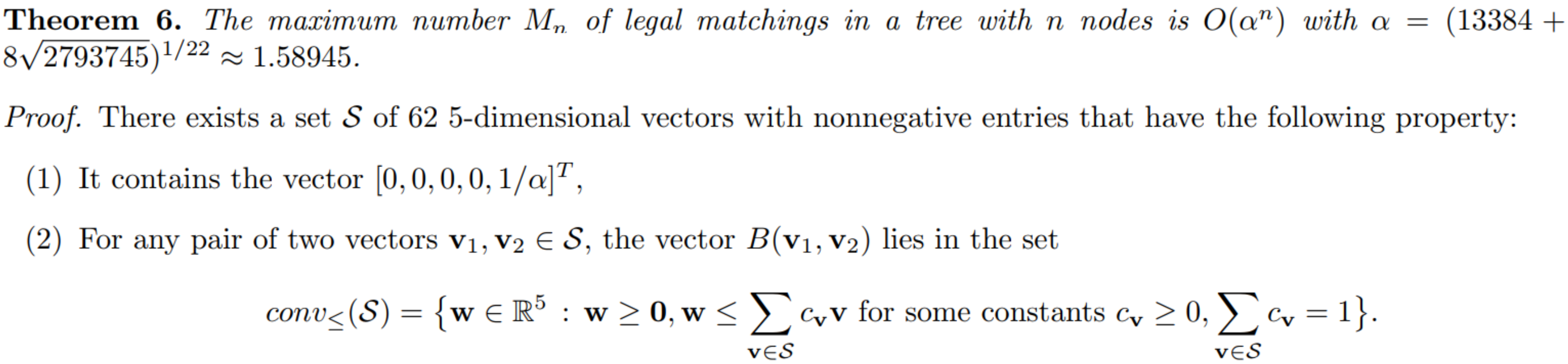
(1) It contains the vector $[0, 0, 0, 0, 1/\alpha]^T$,

(2) For any pair of two vectors $\mathbf{v}_1, \mathbf{v}_2 \in \mathcal{S}$, the vector $B(\mathbf{v}_1, \mathbf{v}_2)$ lies in the set

$$conv_{\leq}(\mathcal{S}) = \left\{\mathbf{w} \in \mathbb{R}^5 : \mathbf{w} \geq \mathbf{0}, \mathbf{w} \leq \sum_{\mathbf{v}\in\mathcal{S}} c_{\mathbf{v}}\mathbf{v} \text{ for some constants } c_{\mathbf{v}} \geq 0, \sum_{\mathbf{v}\in\mathcal{S}} c_{\mathbf{v}} = 1\right\}.$$

Here, the inequalities hold componentwise. Note that $conv_{\leq}(\mathcal{S})$ is a bounded and convex set by construction.





Theorem 6. *The maximum number M_n of legal matchings in a tree with n nodes is $O(\alpha^n)$ with $\alpha = (13384 + 8\sqrt{2793745})^{1/22} \approx 1.58945$.*

Proof. There exists a set $\mathcal{S}$ of 62 5-dimensional vectors with nonnegative entries that have the following property:

(1) It contains the vector $[0, 0, 0, 0, 1/\alpha]^T$,

(2) For any pair of two vectors $\mathbf{v}_1, \mathbf{v}_2 \in \mathcal{S}$, the vector $B(\mathbf{v}_1, \mathbf{v}_2)$ lies in the set

$$conv_{\leq}(\mathcal{S}) = \left\{\mathbf{w} \in \mathbb{R}^5 \,:\, \mathbf{w} \geq \mathbf{0}, \mathbf{w} \leq \sum_{\mathbf{v}\in\mathcal{S}} c_{\mathbf{v}}\mathbf{v} \text{ for some constants } c_{\mathbf{v}} \geq 0, \sum_{\mathbf{v}\in\mathcal{S}} c_{\mathbf{v}} = 1\right\}.$$

The proof subsequently leverages these two properties (1) and (2) to prove by induction that $\alpha^{-n-1}v(T)$ lies in $conv_{\leq}(S)$

- **Sketch of inductive proof that $\alpha^{-n-1}v(T)$ is in $conv_{\leq}(S)$**

- Assume that the left child v_1 has k nodes and the right child v_2 has $n\text{-}k\text{-}1$ nodes.

- $\alpha^{-n-1}v(T) = \alpha^{-n-1} B(\mathrm{v}_1, v_2) = B(\alpha^{-k-1}\mathrm{v}_1, \alpha^{-n+k}\mathrm{v}_2)$

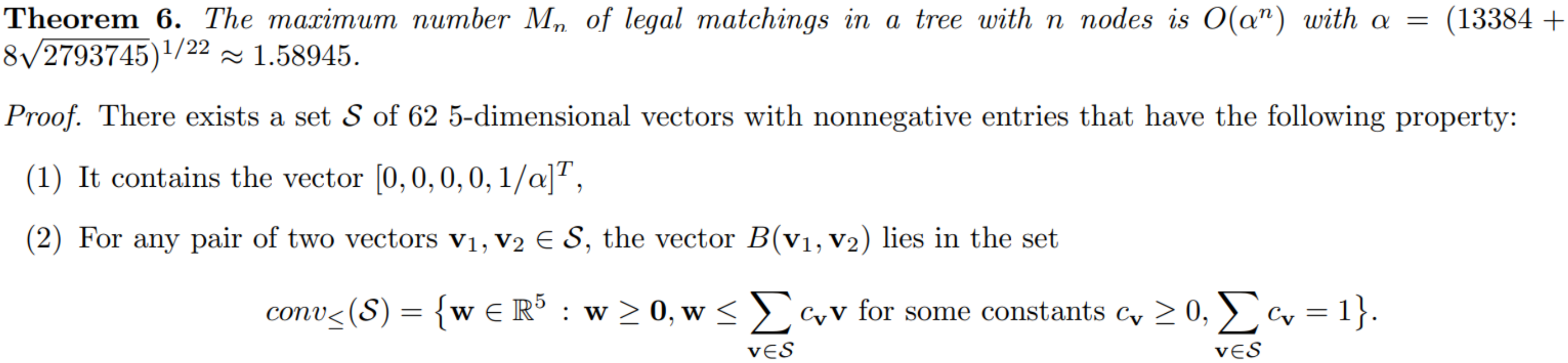
- By induction $\alpha^{-k-1}\mathrm{v}_1$ *and* $\alpha^{-n+k}\mathrm{v}_2$ both lie in $conv_{\leq}(S)$.

- So $\alpha^{-k-1}\mathrm{v}_1$ *and* $\alpha^{-n+k}\mathrm{v}_2$ are both coordinate-dominated by convex sums of the vectors S $=\{s_1, s_2\ldots\}$

- Due to bilinearity of B (and non-negativity) we have that $B(\alpha^{-k-1}\mathrm{v}_1, \alpha^{-n+k}\mathrm{v}_2)$ is coordinate-dominated by the application of B to these two convex sums.

- Also due to bilinearity, the application of B to these two convex sums can then be re-written as a convex sum over $B(s_i, s_j)$ vectors.

- Each $B(s_i, s_j)$ vector is (by definition) in $conv_{\leq}(S)$, and $conv_{\leq}(S)$ is convex, so $\alpha^{-n-1}v(T)$ is in $conv_{\leq}(S)$ □



Theorem 6. *The maximum number M_n of legal matchings in a tree with n nodes is $O(\alpha^n)$ with* $\alpha = (13384 + 8\sqrt{2793745})^{1/22} \approx 1.58945$.

Proof. There exists a set $\mathcal{S}$ of 62 5-dimensional vectors with nonnegative entries that have the following property:

(1) It contains the vector $[0, 0, 0, 0, 1/\alpha]^T$,

(2) For any pair of two vectors $\mathbf{v}_1, \mathbf{v}_2 \in \mathcal{S}$, the vector $B(\mathbf{v}_1, \mathbf{v}_2)$ lies in the set

$$conv_{\leq}(\mathcal{S}) = \{\mathbf{w} \in \mathbb{R}^5 : \mathbf{w} \geq \mathbf{0}, \mathbf{w} \leq \sum_{\mathbf{v} \in \mathcal{S}} c_{\mathbf{v}}\mathbf{v} \text{ for some constants } c_{\mathbf{v}} \geq 0, \sum_{\mathbf{v} \in \mathcal{S}} c_{\mathbf{v}} = 1\}.$$

The proof subsequently leverages these two properties (1) and (2) to prove by induction that $\alpha^{-n-1}v(T)$ lies in $conv_{\leq}(S)$

So $[a_0\alpha^{-n-1}, a_1\alpha^{-n-1}, b_0\alpha^{-n-1}, b_1\alpha^{-n-1}, e\alpha^{-n-1}]$ is bounded…

…so a_0, a_1, b_0, b_1 are $O(\alpha^n)$…

… so $a_0 + a_1 + b_0 + b_1$ = number of legal matchings is $O(\alpha^n)$!

- The recurrence can easily be leveraged to efficiently list these legal matchings, and thus to list relevant convex *X*-colourings; this yields an algorithm for dmp2 with running time O*(1.5895^n).

- Can we bound the number of legal matchings more accurately?

- No! There are trees that have ϴ(1.5895^n) legal matchings, so this is the best we can do with this particular approach:

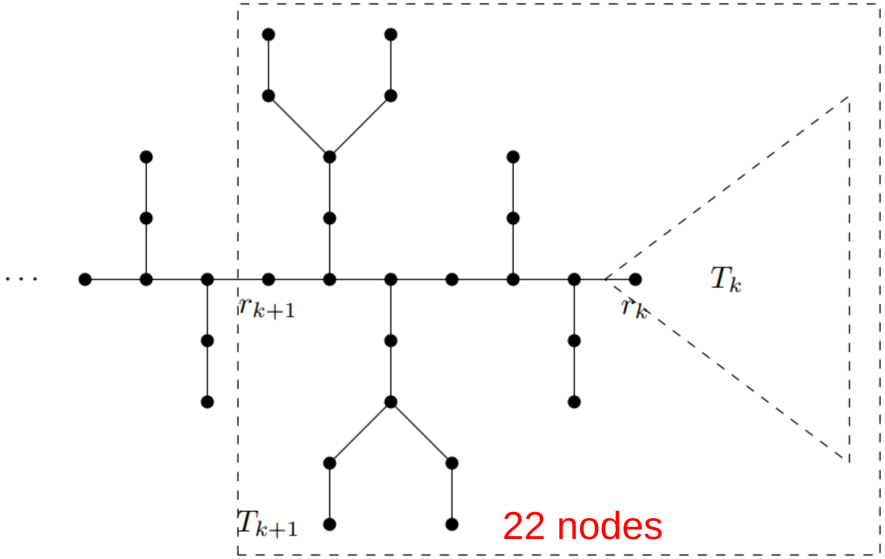
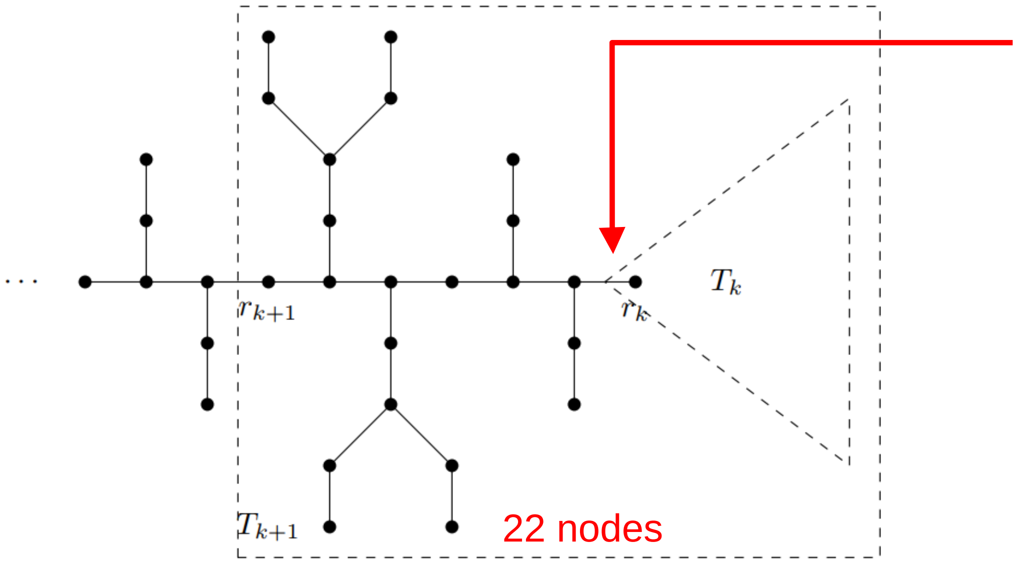


Figure 10: Construction of a sequence of trees with many legal matchings.

- The recurrence can easily be leveraged to efficiently list these legal matchings, and thus to list relevant convex *X*-colourings; this yields an algorithm for dmp2 with running time $O^*(1.5895^n)$.

- Can we bound the number of legal matchings more accurately?

- No! There are trees that have $\Theta(1.5895^n)$ legal matchings, so this is the best we can do with this particular approach:



10144 matchings cover this edge;

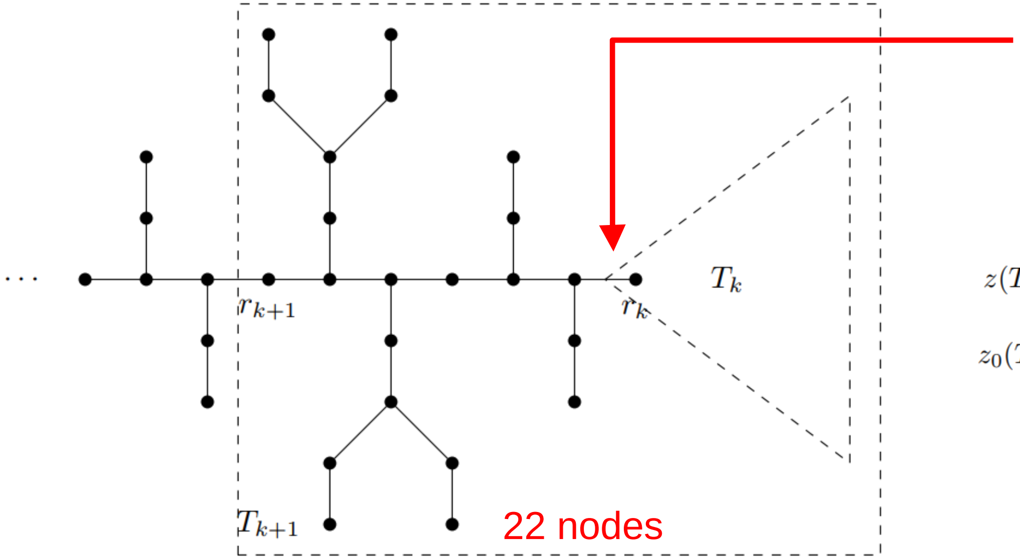
19888 matchings do not cover this edge

Figure 10: Construction of a sequence of trees with many legal matchings.

- The recurrence can easily be leveraged to efficiently list these legal matchings, and thus to list relevant convex *X*-colourings; this yields an algorithm for dmp2 with running time O*(1.5895^n).

- Can we bound the number of legal matchings more accurately?

- No! There are trees that have ϴ(1.5895^n) legal matchings, so this is the best we can do with this particular approach:



10144 matchings cover this edge;

19888 matchings do not cover this edge

$$z(T_{k+1}) = 19888z(T_k) + 10144z_0(T_k).$$

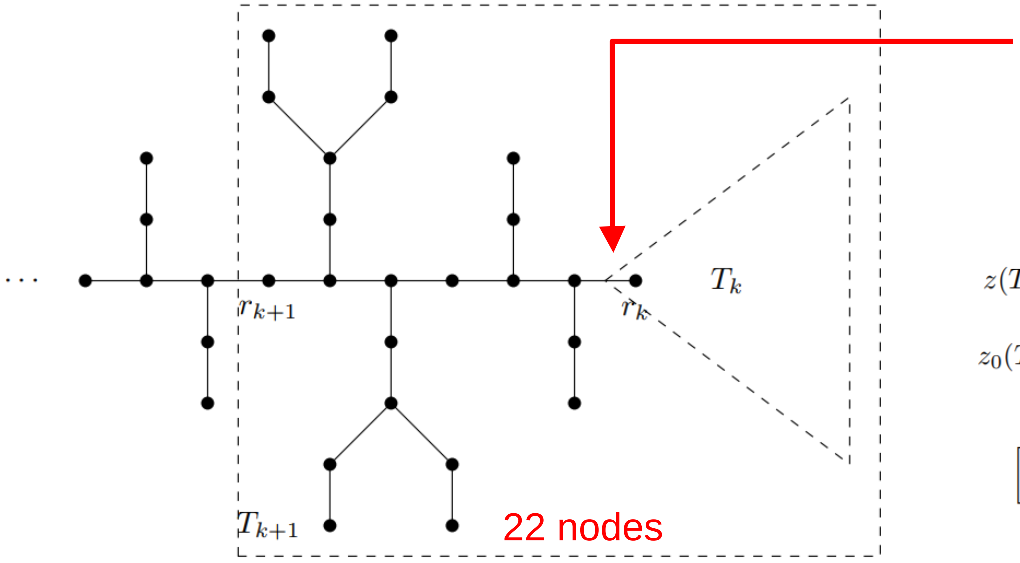
$$z_0(T_{k+1}) = 13456z(T_k) + 6880z_0(T_k).$$

Figure 10: Construction of a sequence of trees with many legal matchings.

- The recurrence can easily be leveraged to efficiently list these legal matchings, and thus to list relevant convex *X*-colourings; this yields an algorithm for dmp2 with running time O*(1.5895^n).

- Can we bound the number of legal matchings more accurately?

- No! There are trees that have ϴ(1.5895^n) legal matchings, so this is the best we can do with this particular approach:



22 nodes

10144 matchings cover this edge;

19888 matchings do not cover this edge

$$z(T_{k+1}) = 19888z(T_k) + 10144z_0(T_k).$$

$$z_0(T_{k+1}) = 13456z(T_k) + 6880z_0(T_k).$$

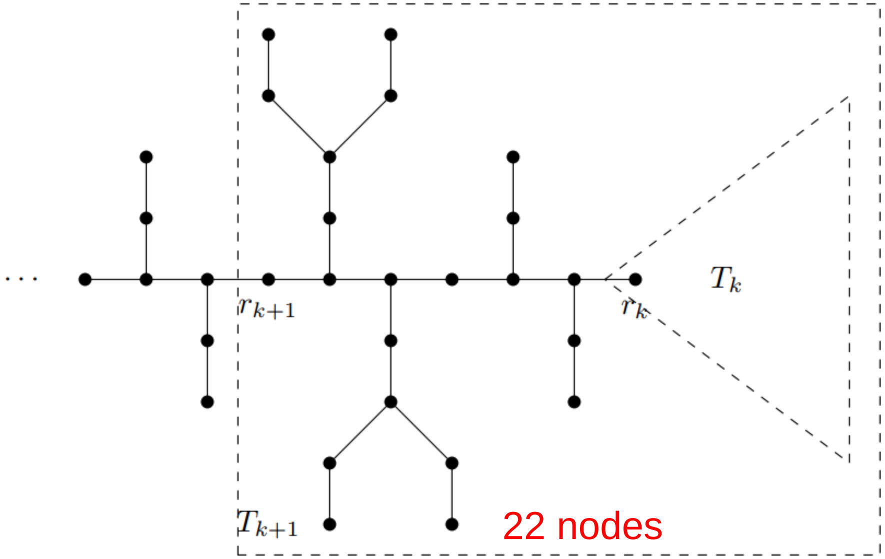
$$\begin{bmatrix} z(T_k) \\ z_0(T_k) \end{bmatrix} = \begin{bmatrix} 19888 & 10144 \\ 13456 & 6880 \end{bmatrix}^k \begin{bmatrix} z(T_0) \\ z_0(T_0) \end{bmatrix}.$$

Figure 10: Construction of a sequence of trees with many legal matchings.

- The recurrence can easily be leveraged to efficiently list these legal matchings, and thus to list relevant convex *X*-colourings; this yields an algorithm for dmp2 with running time O*(1.5895^n).

- Can we bound the number of legal matchings more accurately?

- No! There are trees that have Ө(1.5895^n) legal matchings, so this is the best we can do with this particular approach:



22 nodes

$$z(T_{k+1}) = 19888z(T_k) + 10144z_0(T_k).$$

$$z_0(T_{k+1}) = 13456z(T_k) + 6880z_0(T_k).$$

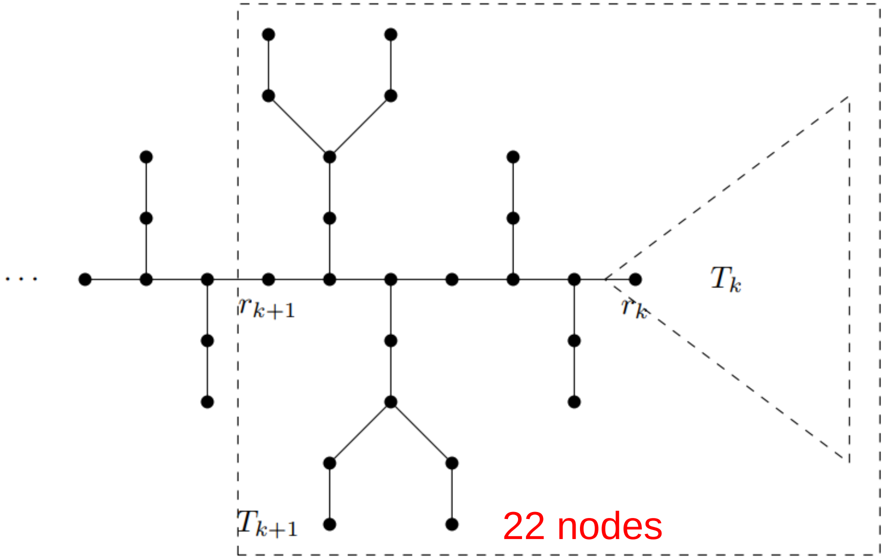
$$\begin{bmatrix} z(T_k) \\ z_0(T_k) \end{bmatrix} = \begin{bmatrix} 19888 & 10144 \\ 13456 & 6880 \end{bmatrix}^k \begin{bmatrix} z(T_0) \\ z_0(T_0) \end{bmatrix}.$$

Figure 10: Construction of a sequence of trees with many legal matchings.

- The recurrence can easily be leveraged to efficiently list these legal matchings, and thus to list relevant convex *X*-colourings; this yields an algorithm for dmp2 with running time O*(1.5895^n).

- Can we bound the number of legal matchings more accurately?

- No! There are trees that have ϴ(1.5895^n) legal matchings, so this is the best we can do with this particular approach:



$$z(T_{k+1}) = 19888z(T_k) + 10144z_0(T_k).$$

$$z_0(T_{k+1}) = 13456z(T_k) + 6880z_0(T_k).$$

$$\begin{bmatrix} z(T_k) \\ z_0(T_k) \end{bmatrix} = \begin{bmatrix} 19888 & 10144 \\ 13456 & 6880 \end{bmatrix}^k \begin{bmatrix} z(T_0) \\ z_0(T_0) \end{bmatrix}.$$

Larger eigenvalue ≈ 1.5895^{22}
Legal matchings ≈ 1.5895^{22k}
k ≈ n/22

Figure 10: Construction of a sequence of trees with many legal matchings.

- Note that in a tree with no adjacent degree-2 nodes, every matching is legal, and every legal matching is (vacuously) a matching.

- **Corollary:** So trees with maximum degree 3 and without adjacent degree-2 nodes, have at most $O(1.5895^n)$ matchings – note here we are talking about normal matchings, not legal matchings.

- This bound is sharp, because the lower bound construction on the previous slide is such a degree-constrained tree (so legal matchings ⇔ matchings).

- This lies between the $O(1.6181^n)$ bound on matchings for general trees, and the $O(1.5538^n)$ bound for trees where all internal nodes have degree 3; new result!

- **Going further…**

- How about eliminating ever larger ‘islands of illegality’? That is, excluding ever-larger families of convex *X*-colourings, that do not help when searching for optimal solutions to dmp2?

- By eliminating slightly larger ‘islands of illegality’ we get a set of vectors in $\mathbb{R}^{13}$ (rather than $\mathbb{R}^5$) and with the help of Mathematica and linear programming, things can also be shown to work out.

- This improves the bound to $O(1.5833^n)$ but everything starts to get rather messy and unwieldy…

- Better than $O(1.5603^n)$ is, in any case, provably not possible, even if all forms of illegality are excluded (construction not shown today).

- **Conclusions and future work**

- We obtained a O*(1.6181^n) and then O*(1.5895^n) algorithm for computing dmp2 on binary phylogenetic trees, using enumeration. Corollary: a new upper bound on the number of matchings in degree-restricted binary trees.

- A 2-colour *X*-colouring might have multiple optimal extensions, and hence the mapping from 2-colour *X*-colourings to convex *X*-colourings is one-to-many. Currently we rediscover such 2-colour *X*-colourings many times, which is pointless. Can we eliminate this waste?

- Is there an elegant way to generate and analyse the recursions as we eliminate ever larger ‘islands of illegality’?

- This is a lot of heavy enumerative combinatorics to obtain a O*(1.5895^n) algorithm for dmp2! Probably better algorithms can be obtained by designing an algorithm that is not simply based on enumeration.

- See also: recent kernelization (FPT) results by Deen et al.

Thank you for listening!

Given this property of $\mathcal{S}$, we can now prove the following by induction on n: for every rooted binary tree T with n nodes, the vector $\alpha^{-n-1}\mathbf{v}(T)$ lies in $conv_{\leq}(\mathcal{S})$. This is trivial for $n = 0$, since we get the vector $[0, 0, 0, 0, 1/\alpha]^T$ for the empty tree, which lies in $\mathcal{S}$ by property (1) and thus in turn in $conv_{\leq}(\mathcal{S})$. For the induction step, we can apply property (2) of $\mathcal{S}$. Assume that the two branches T_1 and T_2 (possibly empty) of T satisfy the statement, and let them have k and $n - k - 1$ nodes respectively. We have

$$\alpha^{-n-1}\mathbf{v}(T) = \alpha^{-n-1}B(\mathbf{v}_1, \mathbf{v}_2) = B(\alpha^{-k-1}\mathbf{v}_1, \alpha^{-n+k}\mathbf{v}_2).$$

By the induction hypothesis, both $\alpha^{-k-1}\mathbf{v}_1$ and $\alpha^{-n+k}\mathbf{v}_2$ lie in $conv_{\leq}(\mathcal{S})$, so there exist linear combinations of the elements of $\mathcal{S}$ with nonnegative coefficients such that

$$\alpha^{-k-1}\mathbf{v}_1 \leq \sum_{\mathbf{v}\in\mathcal{S}} c_{\mathbf{v}}^{(1)}\mathbf{v}$$

and

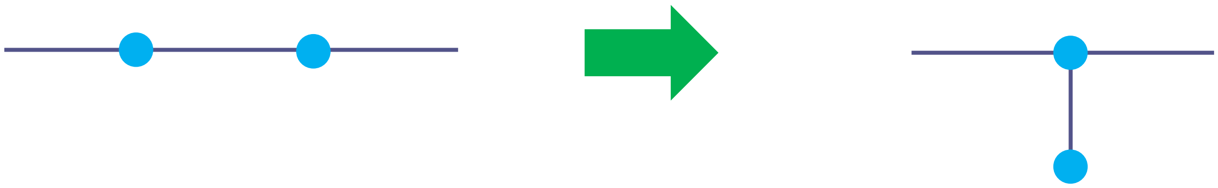
$$\alpha^{-n+k}\mathbf{v}_2 \leq \sum_{\mathbf{v}\in\mathcal{S}} c_{\mathbf{v}}^{(2)}\mathbf{v},$$

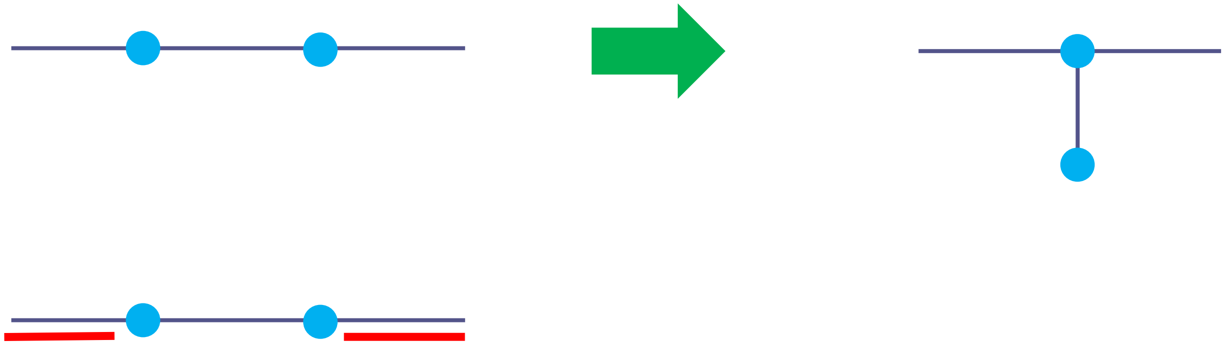
thus by (bi-)linearity of B

$$\begin{aligned}\alpha^{-n-1}\mathbf{v}(T) &= B(\alpha^{-k-1}\mathbf{v}_1, \alpha^{-n+k}\mathbf{v}_2)\\ &\leq \sum_{\mathbf{v}\in\mathcal{S}}\sum_{\mathbf{w}\in\mathcal{S}} c_{\mathbf{v}}^{(1)}c_{\mathbf{w}}^{(2)}B(\mathbf{v}, \mathbf{w}).\end{aligned}$$

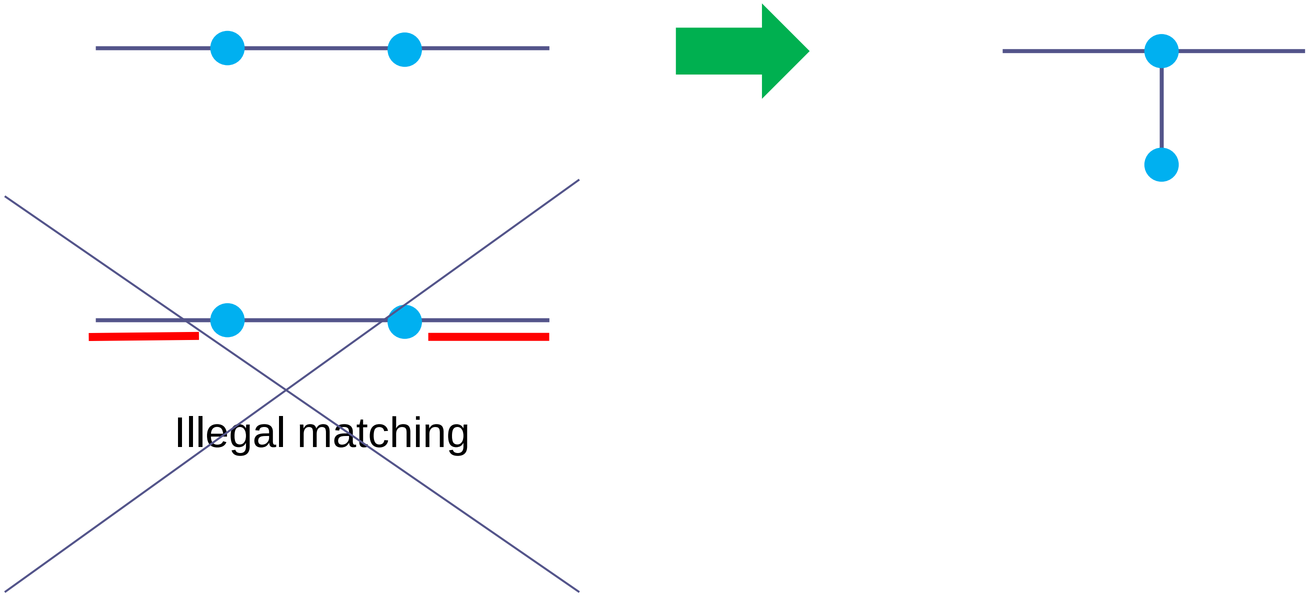
Since $B(\mathbf{v}, \mathbf{w}) \in conv_{\leq}(\mathcal{S})$ for all $\mathbf{v}$ and $\mathbf{w}$ and $conv_{\leq}(\mathcal{S})$ is convex, it follows that $\alpha^{-n-1}\mathbf{v}(T) \in conv_{\leq}(\mathcal{S})$, completing the induction.

In particular, we have shown that the entries of the vector $\alpha^{-n-1}\mathbf{v}(T)$ are bounded. The total number of legal matchings of T is the sum of the entries of $\mathbf{v}(T)$, so it follows that this number is $O(\alpha^n)$. □

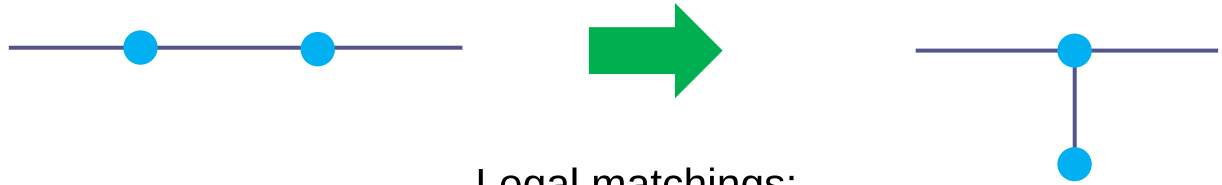




Illegal matching



Illegal matching



Legal matchings: