

Crazy bijections between

- planar maps
- β -trees
- perms

Joint work with Sergay Kitav
Pavel Salimov
Chris Severs

We start with the $\beta(1,0)$ -trees

- * rooted
- * labelled
- * ordered

All trees w 1 edge

All trees w 2 edges

All trees w 3 edges

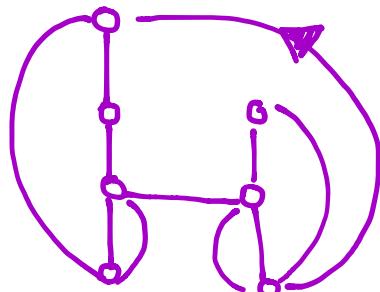
All trees w 4 edges

Why should we care about these objects

- * Trees are a central object in CS
- * ... and in combinatorics
- * but why this type?

↳ because they "encode" planar maps
and in fact they were created for that purpose by Cori, Jacquot and Schaeffer

BUT WHAT ARE PLANAR MAPS?



your
favorite
planar
map

By "maps" I really mean graphs, drawn in a plane, that are

- * rooted
- * non-separable
- * edges only cross at vertices

All maps w 2 edges

All maps w 3 edges

Why should we care about these objects?

* pop up in quantum gravity

statistical mechanics

topology of surfaces

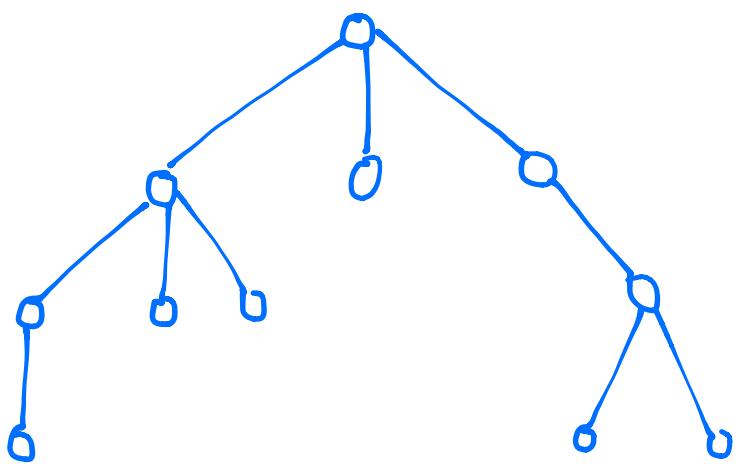
...

* I care about them because I find them
beautiful!

Now, there is a bijection from the trees to
the maps.

Basic idea: Start at the leaves and work your
way up to the root

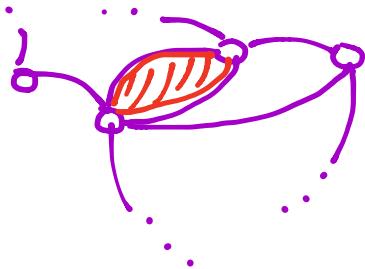
Here's your favorite tree



Now (finally) the main question:

How are properties of maps transferred
to properties of trees?

Focus on 2-faces in the maps



Let's look at the example we did above...

Conjecture: 2-faces in the maps



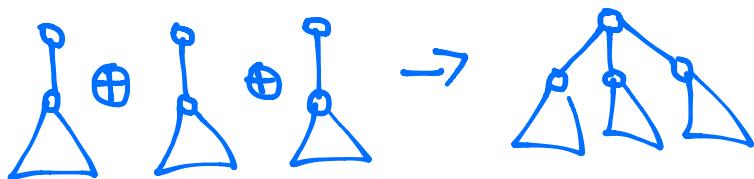
single children w/max label

This happens to be true

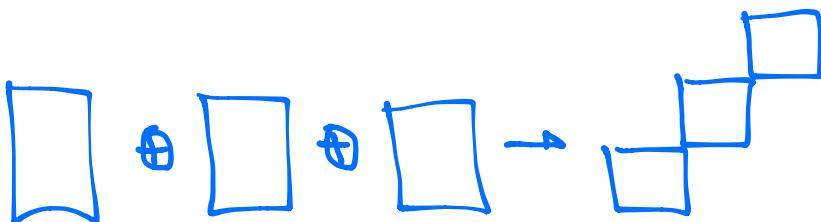
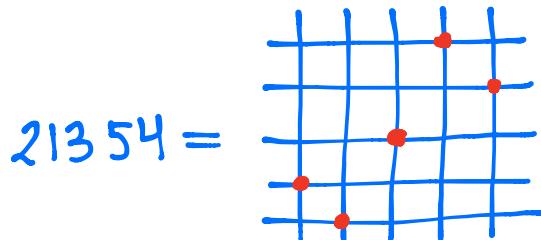
What about the permutations?

Claesson, Kitaev and Steingrímsson created an indirect bijection by showing that the trees have the same generation rules as a subclass of permutations.

Rule 1: Decomp. from indecomp.



If we draw perms



Example

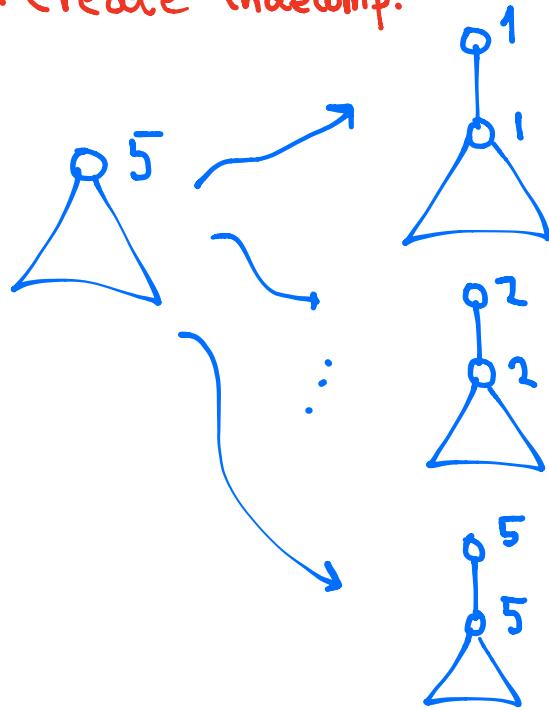
21

1

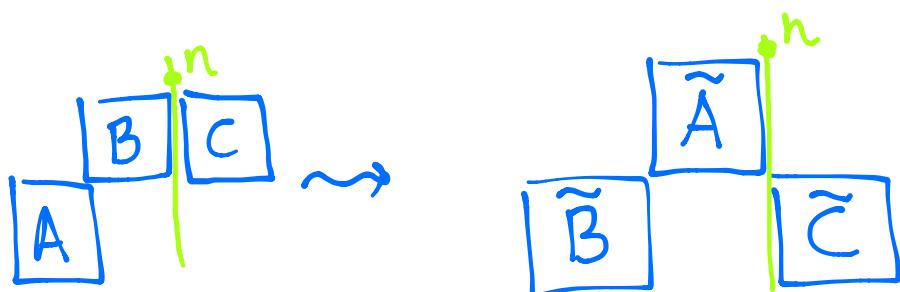
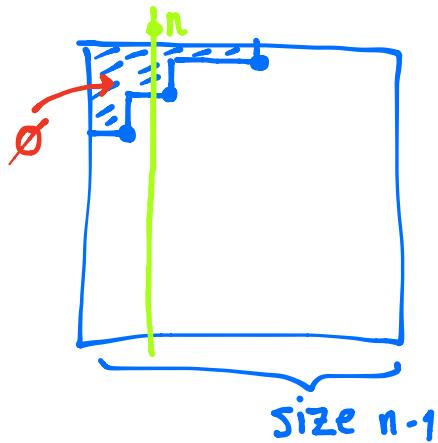
21

21
21

Rule 2: Create indecomp.

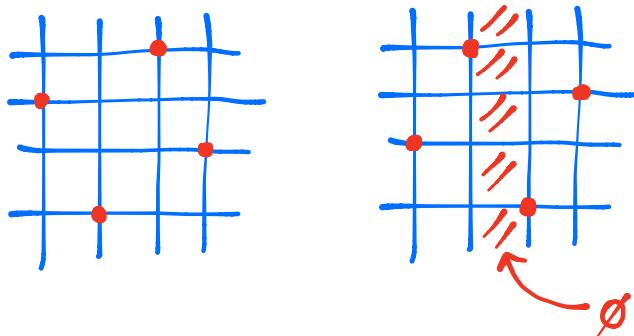


What do we do with permutations?



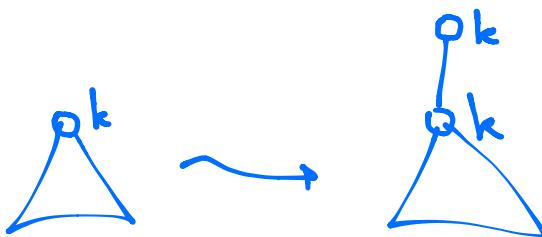
What subclass of perms has these generation rules?

↳ the avoiders of

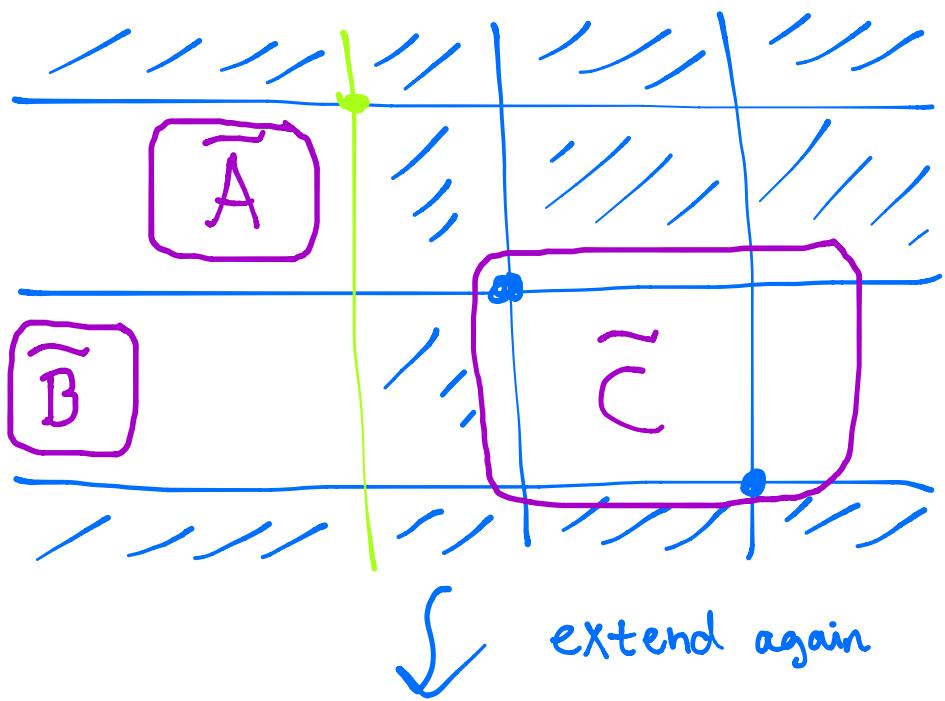
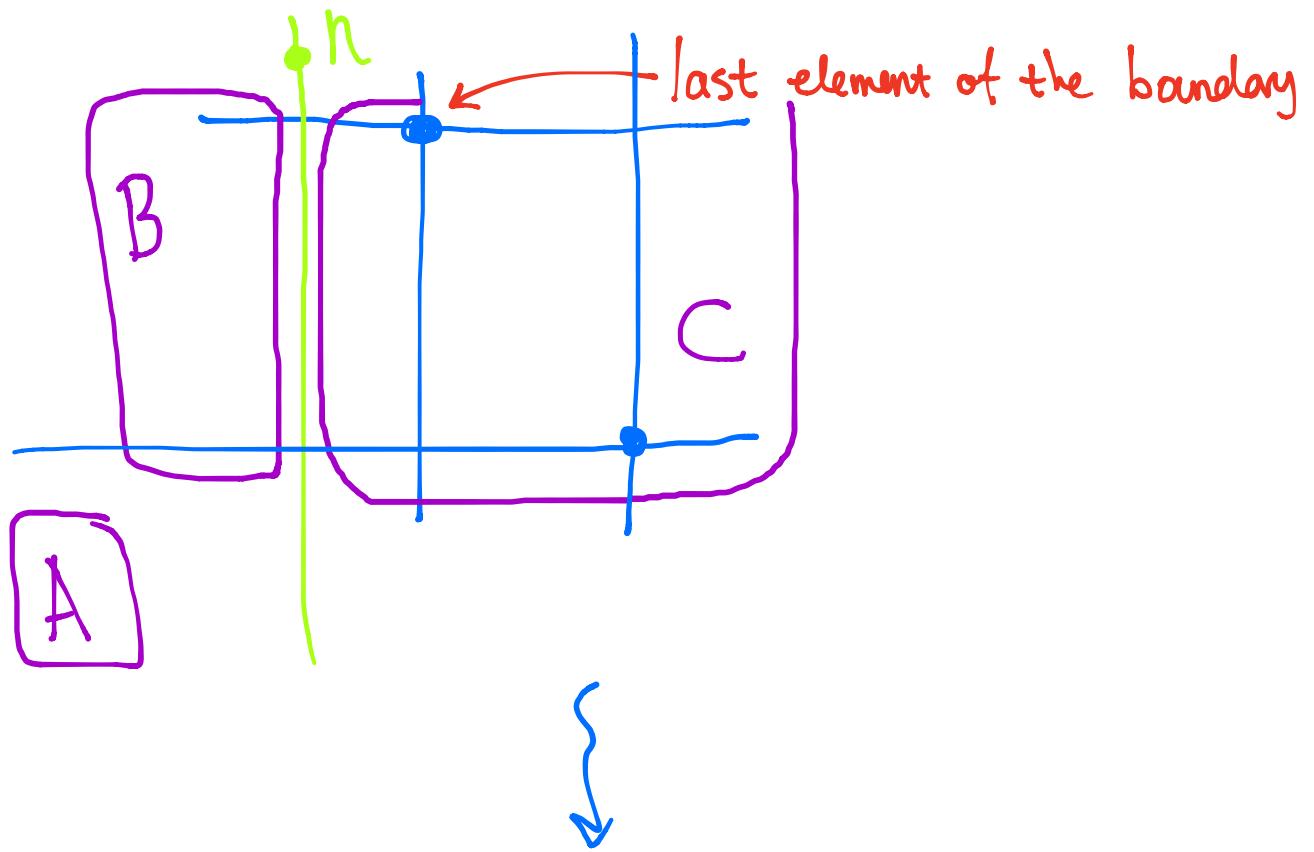


What about those 2-faces and the single children w/max label

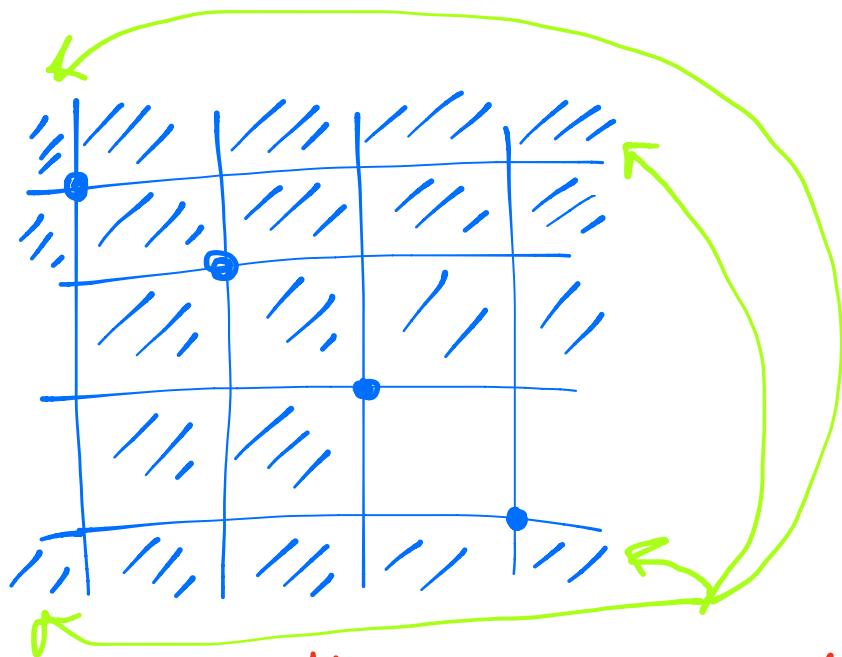
Main idea: a single child w/max label somewhere inside a tree began as a single child of a root w/max label



This corresponds to placing n as deep as we can inside the perm



extend again



If more extensions / additions
take place then we lose some
shadings

This is the case
where we extend
as far in as possible

If you consider all cases
then you get 8 mesh
patterns

