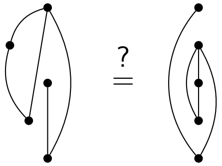
Normal forms for planar string diagrams

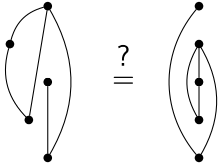
Antonin Delpeuch, Jamie Vicary

SYCO 2

Background: word problem in higher categories



Background: word problem in higher categories

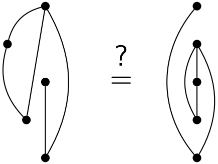


Theorem (Makkai, 2005)

The word problem for cells of a finitely generated strict n-category is decidable.

Idea of the proof: the configuration space of a given diagram is finite.

Background: word problem in higher categories



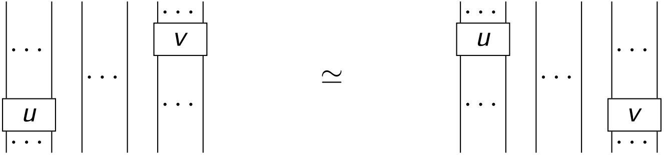
Theorem (Makkai, 2005)

The word problem for cells of a finitely generated strict n-category is decidable.

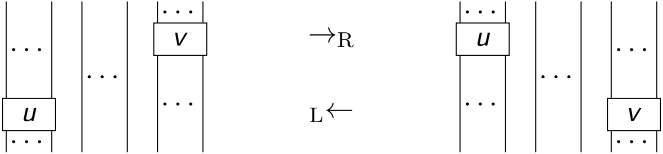
Idea of the proof: the configuration space of a given diagram is finite.

As an algorithm, this is vastly inefficient.

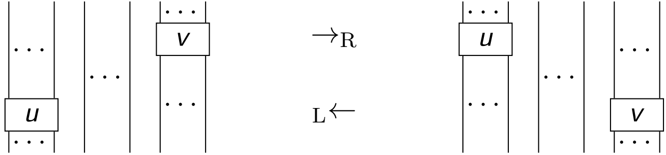
The planar case



The planar case



The planar case



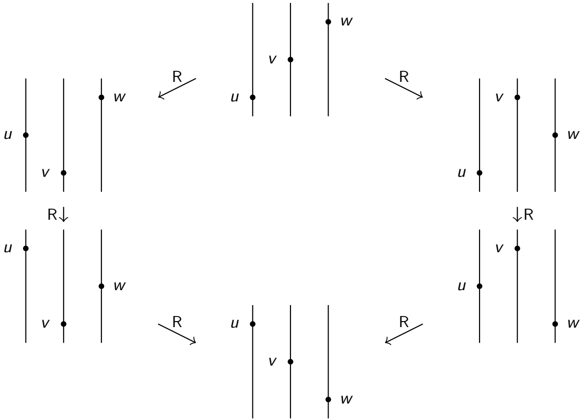
Theorem

$\rightarrow_R$ is convergent (terminating and confluent) on connected diagrams.

Confluence of right exchanges

Lemma

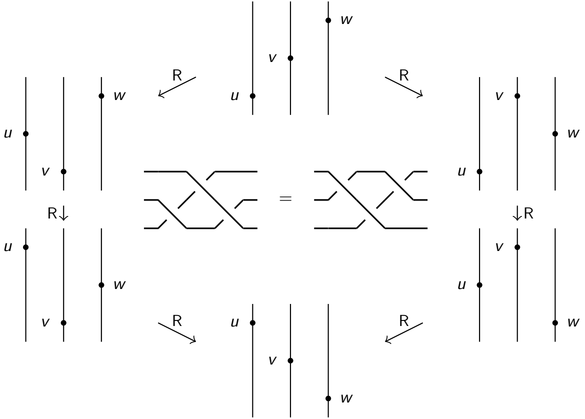
$\rightarrow_R$ is locally confluent.



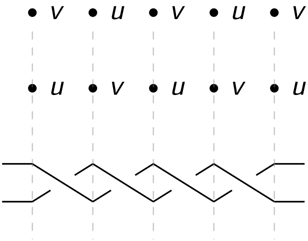
Confluence of right exchanges

Lemma

$\to_R$ is locally confluent.



Termination



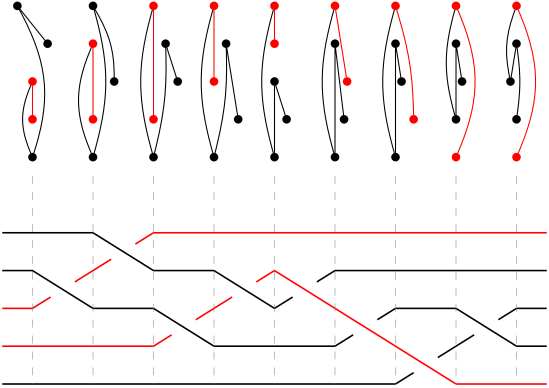
Termination fails in general, but:

Theorem

$\to_R$ terminates in $O(n^3)$ for connected diagrams of size n.

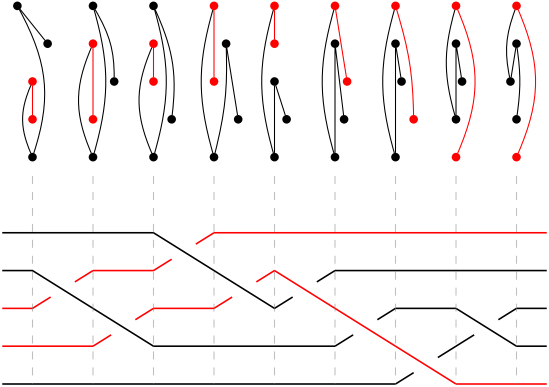
Proof of termination

First, in the case of linear diagrams:



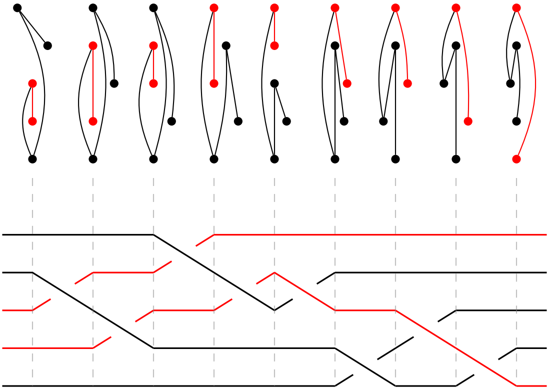
Proof of termination

First, in the case of linear diagrams:



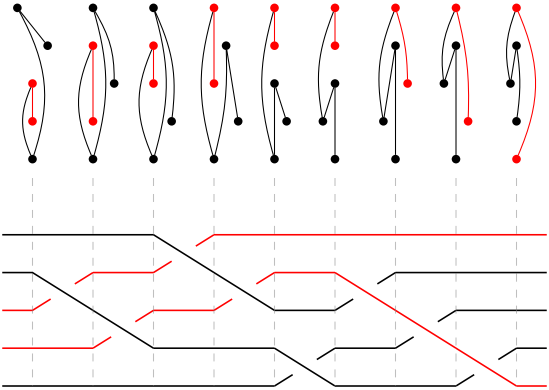
Proof of termination

First, in the case of linear diagrams:



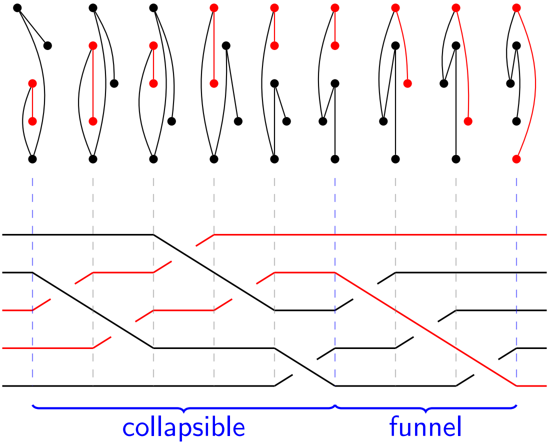
Proof of termination

First, in the case of linear diagrams:



Proof of termination

First, in the case of linear diagrams:



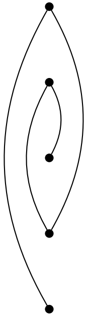
Upper bound on derivation length

From this decomposition, we obtain inductively the following bound:

Lemma

$\to_R$ *terminates on linear graphs of length* n *in* $O(n^3)$.

The bound is attained for spiral-shaped diagrams:



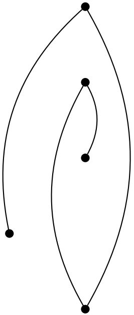
Upper bound on derivation length

From this decomposition, we obtain inductively the following bound:

Lemma

$\rightarrow_R$ *terminates on linear graphs of length* n *in* $O(n^3)$.

The bound is attained for spiral-shaped diagrams:



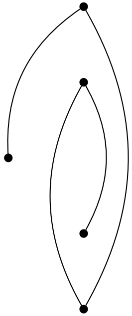
Upper bound on derivation length

From this decomposition, we obtain inductively the following bound:

Lemma

$\rightarrow_R$ *terminates on linear graphs of length n in* $O(n^3)$.

The bound is attained for spiral-shaped diagrams:



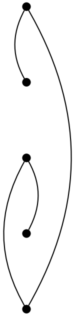
Upper bound on derivation length

From this decomposition, we obtain inductively the following bound:

Lemma

$\rightarrow_R$ *terminates on linear graphs of length* n *in* $O(n^3)$.

The bound is attained for spiral-shaped diagrams:



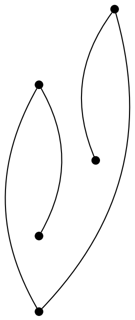
Upper bound on derivation length

From this decomposition, we obtain inductively the following bound:

Lemma

$\rightarrow_R$ terminates on linear graphs of length n in $O(n^3)$.

The bound is attained for spiral-shaped diagrams:



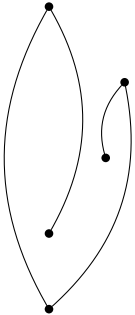
Upper bound on derivation length

From this decomposition, we obtain inductively the following bound:

Lemma

$\rightarrow_R$ *terminates on linear graphs of length n in* $O(n^3)$.

The bound is attained for spiral-shaped diagrams:



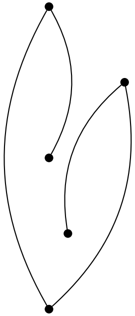
Upper bound on derivation length

From this decomposition, we obtain inductively the following bound:

Lemma

$\rightarrow_R$ *terminates on linear graphs of length* n *in* $O(n^3)$.

The bound is attained for spiral-shaped diagrams:



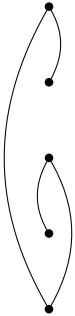
Upper bound on derivation length

From this decomposition, we obtain inductively the following bound:

Lemma

$\rightarrow_R$ terminates on linear graphs of length n in $O(n^3)$.

The bound is attained for spiral-shaped diagrams:



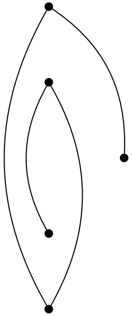
Upper bound on derivation length

From this decomposition, we obtain inductively the following bound:

Lemma

$\rightarrow_R$ *terminates on linear graphs of length n in* $O(n^3)$.

The bound is attained for spiral-shaped diagrams:



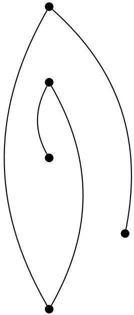
Upper bound on derivation length

From this decomposition, we obtain inductively the following bound:

Lemma

$\to_R$ terminates on linear graphs of length n in $O(n^3)$.

The bound is attained for spiral-shaped diagrams:



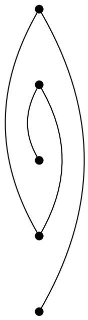
Upper bound on derivation length

From this decomposition, we obtain inductively the following bound:

Lemma

$\to_R$ *terminates on linear graphs of length* n *in* $O(n^3)$.

The bound is attained for spiral-shaped diagrams:



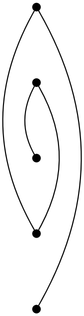
Upper bound on derivation length

From this decomposition, we obtain inductively the following bound:

Lemma

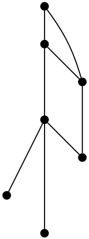
$\rightarrow_R$ terminates on linear graphs of length n in $O(n^3)$.

The bound is attained for spiral-shaped diagrams:



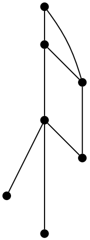
Number of steps for a spiral with n vertices: $\binom{n}{3}$

General case

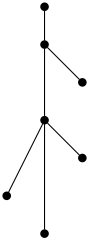


Connected graph G

General case

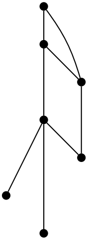


Connected graph G

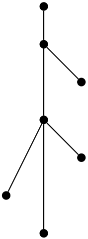


Spanning tree G'

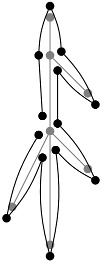
General case



Connected graph G

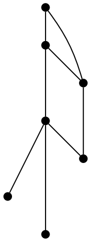


Spanning tree G'

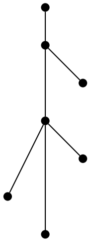


Linear envelope

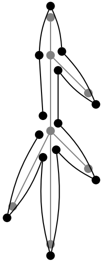
General case



Connected graph G



Spanning tree G'

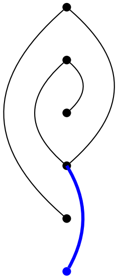


Linear envelope

$O(n^3)$ exchanges, each of them taking $O(n)$ time to perform: word problem solved in $O(n^4)$.

Direct algorithm to compute normal forms

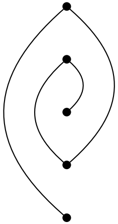
By induction on the number of edges.



First case: the diagram has a leaf.

Direct algorithm to compute normal forms

By induction on the number of edges.

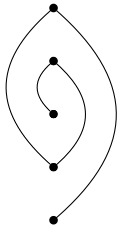


First case: the diagram has a leaf.

- remove this leaf;

Direct algorithm to compute normal forms

By induction on the number of edges.

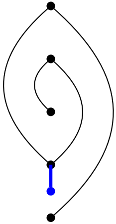


First case: the diagram has a leaf.

- remove this leaf;
- normalize the diagram recursively;

Direct algorithm to compute normal forms

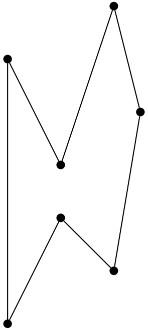
By induction on the number of edges.



First case: the diagram has a leaf.

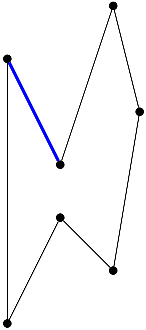
- remove this leaf;
- normalize the diagram recursively;
- add the leaf back at the unique height making the diagram normalized.

Direct algorithm to compute normal forms



Second case: the diagram has a face

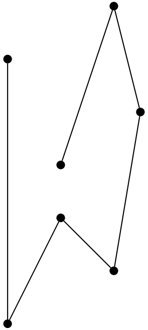
Direct algorithm to compute normal forms



Second case: the diagram has a face

- Find an *eliminable edge* in the face and remove it;

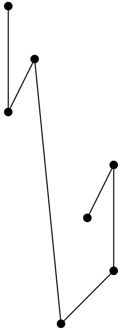
Direct algorithm to compute normal forms



Second case: the diagram has a face

- Find an *eliminable edge* in the face and remove it;

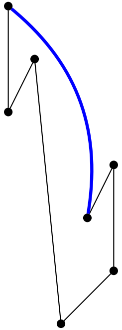
Direct algorithm to compute normal forms



Second case: the diagram has a face

- Find an *eliminable edge* in the face and remove it;
- normalize the graph recursively;

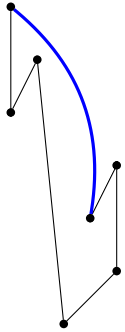
Direct algorithm to compute normal forms



Second case: the diagram has a face

- Find an *eliminable edge* in the face and remove it;
- normalize the graph recursively;
- add the edge back.

Direct algorithm to compute normal forms



Second case: the diagram has a face

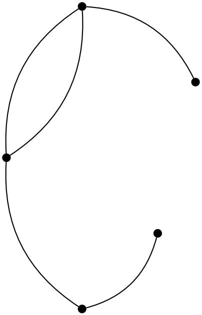
- Find an *eliminable edge* in the face and remove it;
- normalize the graph recursively;
- add the edge back.

Each step removes one edge and requires linear time in the number of vertices, so word problem solved in $O(nm)$.

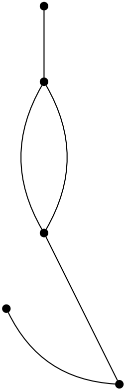
Linear time solution

Theorem

Isotopy of connected planar maps can be decided in linear time (Hopcroft and Wong, 1974)



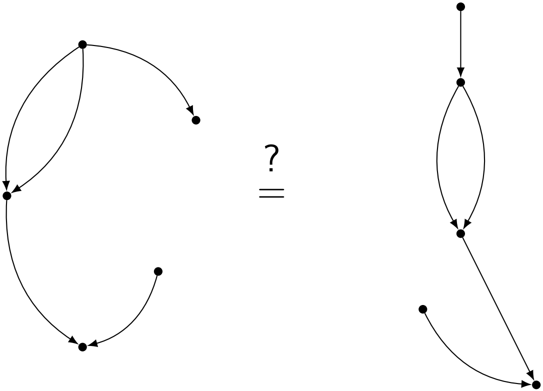
$\stackrel{?}{=}$



Linear time solution

Theorem

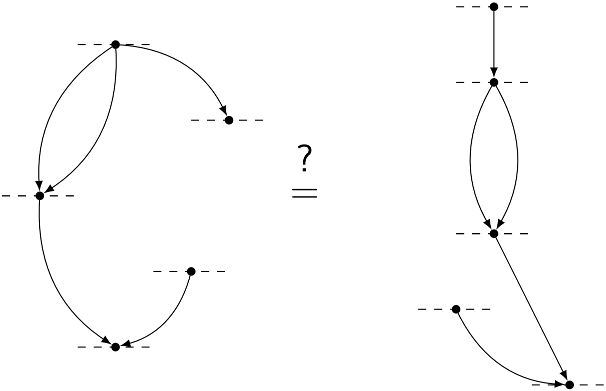
Isotopy of connected planar **directed** *maps can be decided in linear time.*



Linear time solution

Theorem

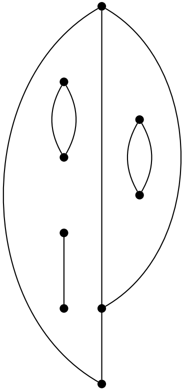
Isotopy of connected **string diagrams** *can be decided in linear time.*



Disconnected case

Theorem

Isotopy of string diagrams can be decided in quadratic time.



References

A. Delpeuch and J. Vicary.
Normal forms for planar connected string diagrams.
ArXiv e-prints, April 2018.

Michael Makkai.
The word problem for computads.
Available on the author's web page http://www.math.mcgill.ca/makkai, 2005.