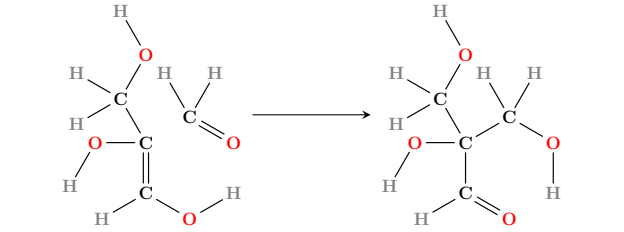
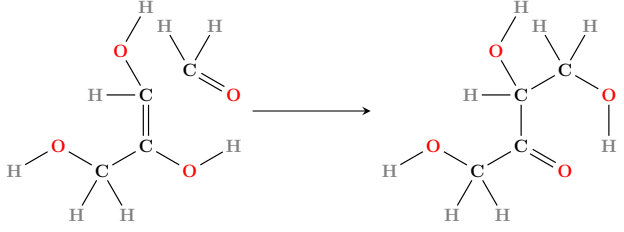
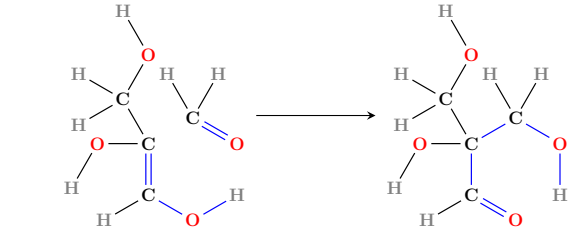
Categories for Chemically Inspired Graph Transformation

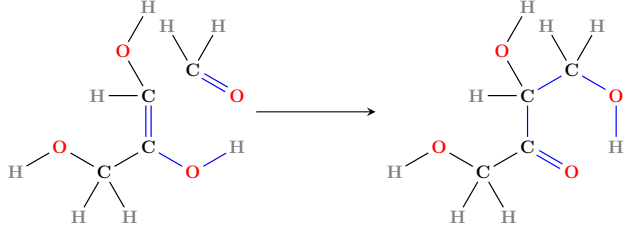
Jonas Stisen

February 2024

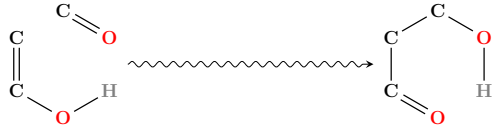


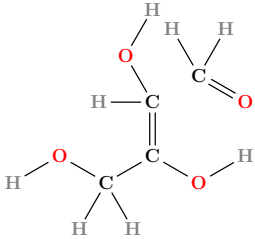




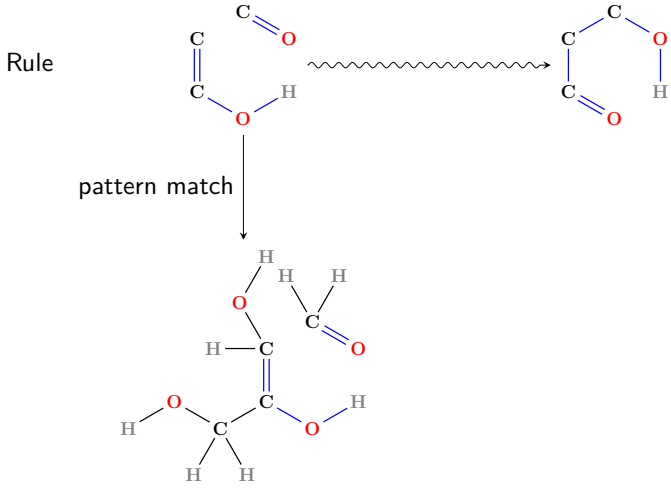


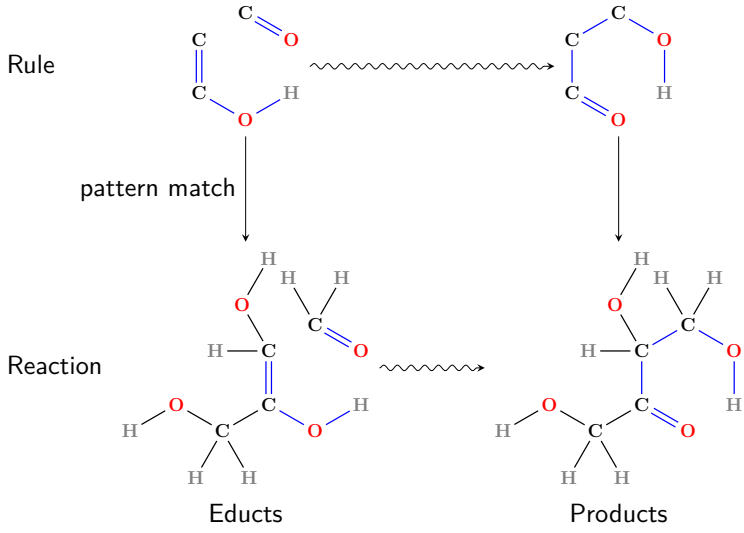
Rule





Educts


Rule
pattern match
Educts


Rule
pattern match
Reaction
Educts
Products

Motivation

We want

- Chemical graph transformation,
- With wildcards (‘any’, ‘one of’, ‘none of’),
- Stereochemistry.

Current state

We have (in MØD)

- Chemical graph transformation,
- With wildcards (‘any’, ‘one of’, ‘none of’),
- Some stereochemistry,
- *But*: Slightly wobbly theoretical foundation.

Categories

- A *category* consists of *objects* (say, graphs) and *morphisms* (say, graph homomorphisms) between objects
- Morphisms compose associatively and an *identity morphism* exists
- ‘Graph gluing’ is generalised by the categorical concept of *pushouts*

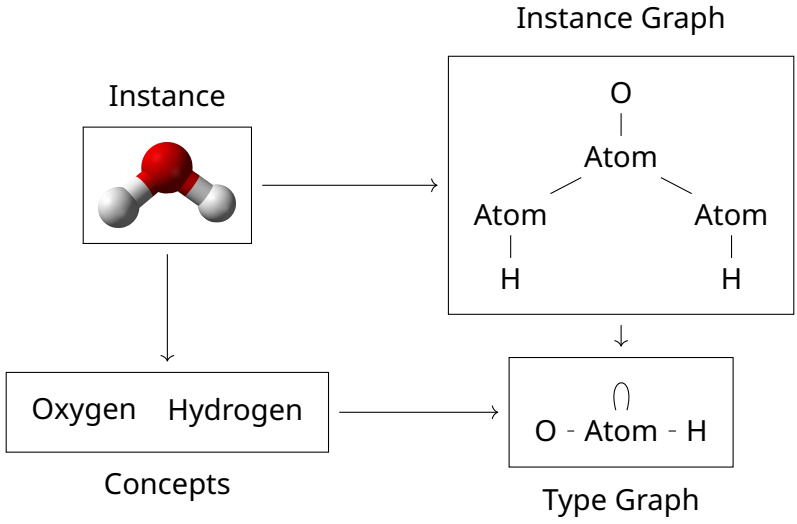
Adhesive Categories

- The original work on algebraic graph transformation treated concrete categories
- We wish to generalise to other categories
- How do characterise the categories that are ‘nice’?
- A hierarchy of *adhesive* categories

Typed Graphs

- A *type graph* is a distinguished graph
- Given a type graph, TG, a *typed graph* is a pair $(G, \tau : G \to \mathrm{TG})$
- We define $\mathbf{uGraph}_{\mathrm{TG}}$ as the category of undirected multigraphs with loops permitted typed over some type graph TG

Typed Graphs



Application Conditions

- **uGraph**$_{TG}$ is problematic in that it allows multi-edges and loops
- *Application conditions* mitigates this problem
- Application conditions restrict what matching morphisms are permissible
- Application conditions can be nested, negated, and combined by conjunction

Application Conditions

- Wildcards by negation and/or conjunction
- Guard against applications resulting in chemically invalid states

Typed Attributed Graphs

- *Data vertices* connected to vertices and edges
- Morphisms perform computation on data nodes
- We denote the category of attributed multigraphs with loops permitted typed over an attributed graph as $\mathbf{AGraph}_{\mathrm{ATG}}$
- Idea: attach first order terms to vertices and edges and perform unification with morphisms

Typed Attributed Graphs

- Typed Attributed Graphs only exist in directed form
- Application conditions don’t work

Thank You For Your Time