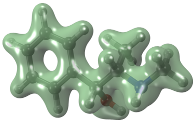
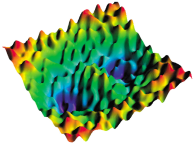
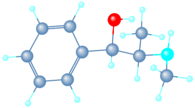
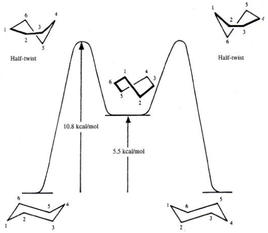
Levels of Abstraction in Computational Chemistry

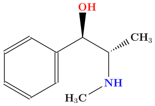




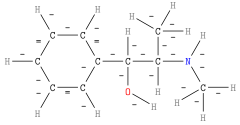
Potential energy surface







Reaction coordinate

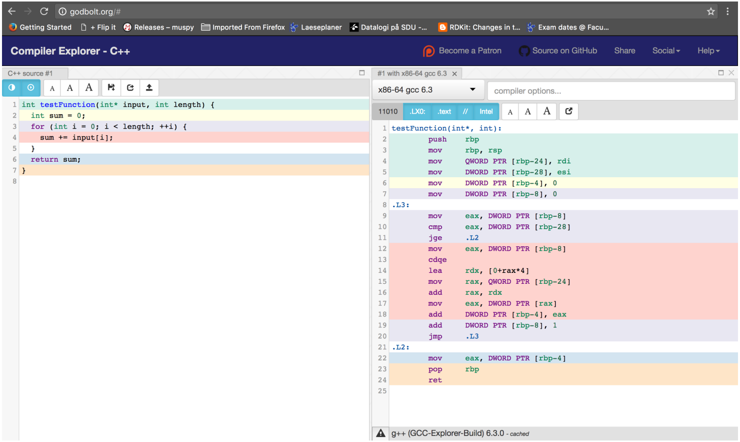


$L \leftarrow K \rightarrow R$

Graph grammar

[Andersen et al., Proceedings of the Royal Society A, 2017]

Levels of Abstraction in Programming



Declarative Description ↔ DSL ↔ C++ ↔ Assembler

Levels of Abstraction in Computer Science

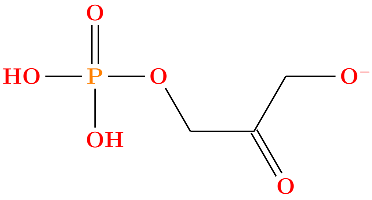
"The psychological profiling [of a Computer Scientist] is mostly the ability to shift levels of abstraction, from low level to high level. To see something in the small and to see something in the large."

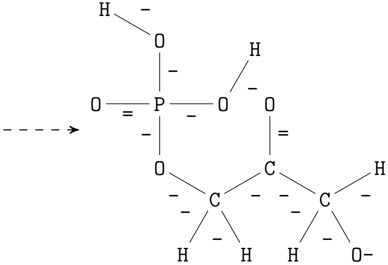
Donald Knuth

Modelling and Analysis of Chemical Systems

Modelling and Analysis of Chemical Systems

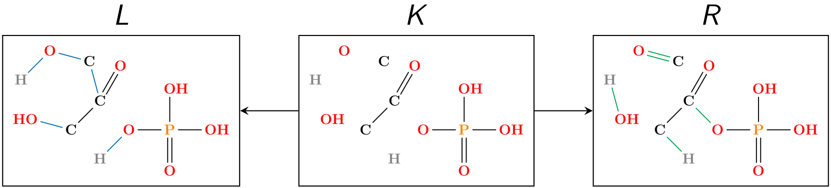
1. Model molecules as labelled graphs.
   - An old idea: [J. J. Sylvester, *Chemistry and Algebra*, Nature 1878]
   - Molecule: simple, connected, labelled graph.
   - Vertex labels: atom type, charge.
   - Edge labels: bond type.





Modelling and Analysis of Chemical Systems

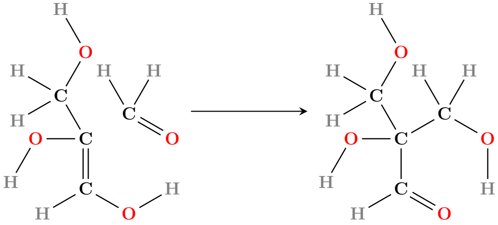
2. Model reaction types and graph transformation rules.

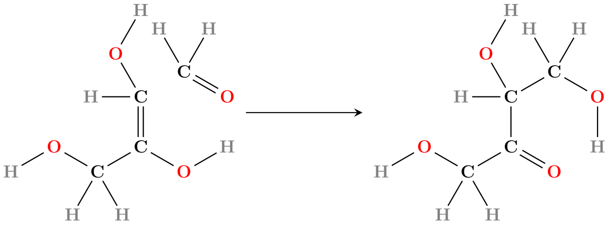


Example: Carbon rearrangement

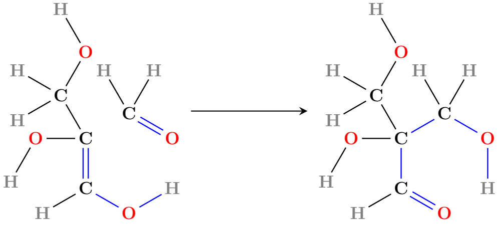
- Aldolase: ketone + aldehyde $\longrightarrow$ ketone
- Aldose-Ketose: aldehyde $\longrightarrow$ ketone
- Ketose-Aldose: ketone $\longrightarrow$ aldehyde
- Phosphohydrolase: H_2O+CnP $\longrightarrow$ Cn+Pi
- Phosphoketolase Pi+ketone $\longrightarrow$ carbonyl + CnP+water
- Transaldolase: Cn+Cm$\longrightarrow$ C(n+3)+C(m-3)
- Transketolase: Cn+Cm$\longrightarrow$ C(n+2)+C(m-2)

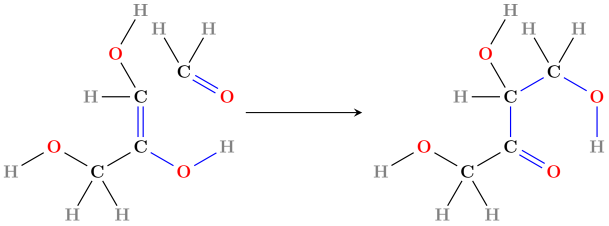
Chemical Reactions (Educts → Products)





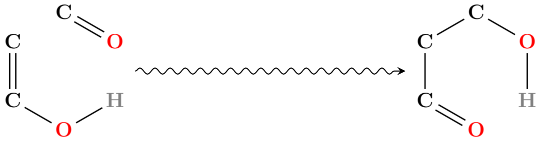
Chemical Reactions (of the Same Type)

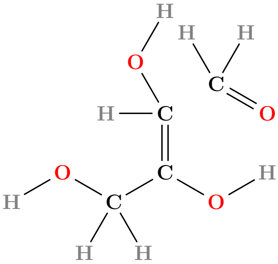




Chemical Reaction Patterns

Rule

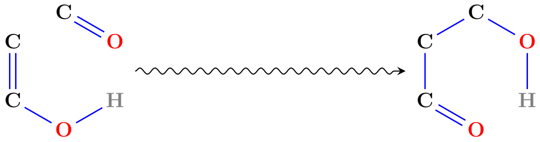




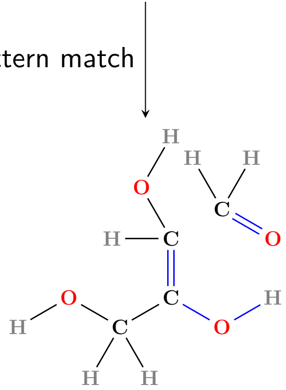
Educts

Chemical Reaction Patterns

Rule

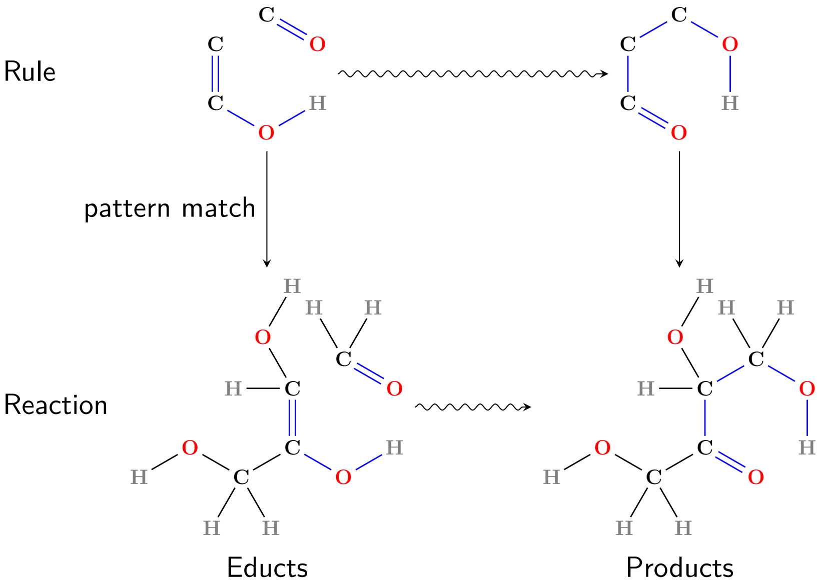


pattern match



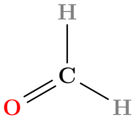
Educts

Chemical Reaction Patterns

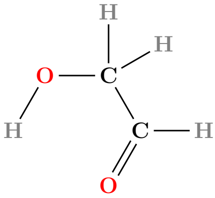


Grammar Example: The Formose Chemistry

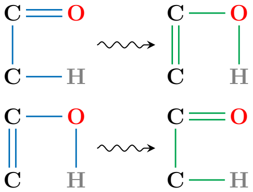
Formaldehyde:



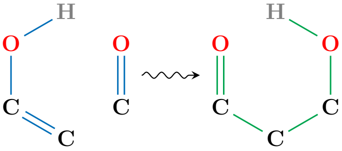
Glycolaldehyde:



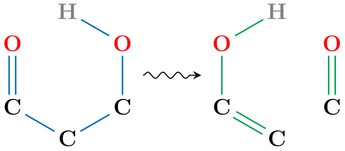
Keto-enol tautomerism:



Aldol addition:



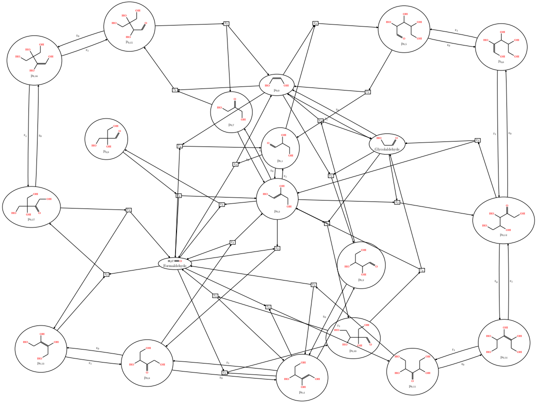
Retro aldol addition:



Modelling and Analysis of Chemical Systems

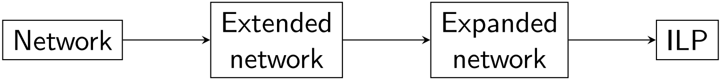
3. Generate a reaction network.

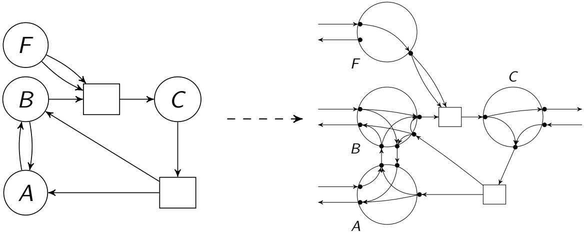
```
dg = dgRuleComp(inputGraphs,
    addSubset(inputGraphs) >> rightPredicate[
        lambda d: all(countCarbon(a) <= 5 for a in d.right)
    ](      repeat(inputRules)    )
)
dg.calc()
```



Modelling and Analysis of Chemical Systems

4. Set up pathway model.





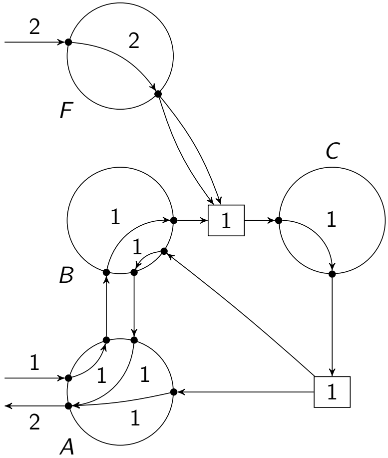
Conservation constraints:

$$\sum_{e\in\delta^+_{\widetilde{E}}(v)} m_v(e^+)f(e) - \sum_{e\in\delta^-_{\widetilde{E}}(v)} m_v(e^-)f(e) = 0 \qquad \forall v \in \widetilde{V}$$

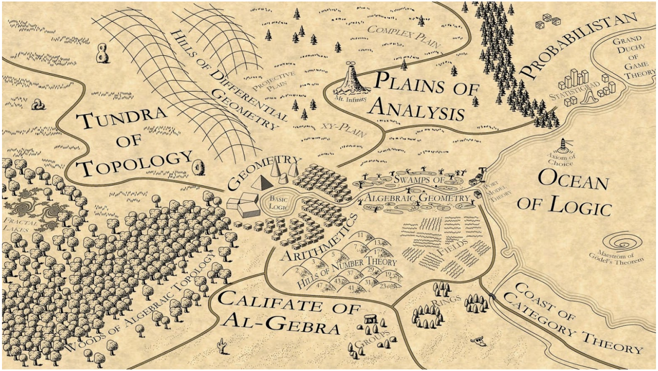
Modelling and Analysis of Chemical Systems

5. Formulate pathway question.

Example: Given 2 formaldehyde and 1 glycolaldehyde, how can 2 glycolaldehyde be produced through autocatalysis.



Category Theory: Mathematistan



Daniel: quite nice. though wrong, as category theory probably is not just a region

Jakob: hehe, ye, category theory is when you drank a too much wine, look at the map, and it suddenly it says “category theory” all over

[Figure by Martin Kuppe]

A Chemical Graph Transformation System

Objects:

- Molecule graph
- Molecule collection
- Pattern match
- Transformation rule
- Reaction network
- ...

Operations:

- Substructure search
- Molecule equivalence
- Rule application
- Reaction network generation
- Isotope tracing
- ...

Fundamental operation: composition of transformation rules
Mathematical framework: category theory

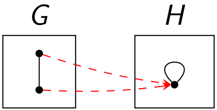
Categories

A category **C**:

- A class of objects: $Ob(\mathbf{C})$
  E.g., connected, labelled graphs.
- A class of morphisms: $Mor(\mathbf{C})$
  E.g., graph monomorphisms, with label constraints.
- An associative morphism composition operator: $\circ$

Graph Morphisms

Def. graph morphism: $m\colon G \to H$ with
$\forall e = (u, v) \in E_G : m(e) = (m(u), m(v)) \in E_H$.
NP-complete (e.g., reduce from Graph Colouring).

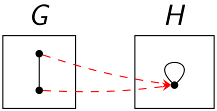


(a) A morphism.

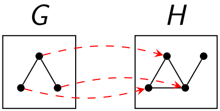
Graph Morphisms

Def. graph monomorphism: an injective graph morphism
i.e., $\forall u, v \in V_G, u \neq v \Rightarrow m(u) \neq m(v)$.

NP-complete (e.g., reduce from HAMILTONIAN CYCLE).



(a) A morphism.



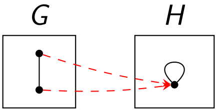
(b) A monomorphism.

Graph Morphisms

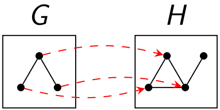
Def. subgraph isomorphism: a graph monomorphism with

$$(u, v) \in E_G \Leftrightarrow (m(u), m(v)) \in E_H.$$

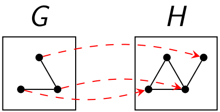
NP-complete (e.g, reduce from CLIQUE).



(a) A morphism.



(b) A monomorphism.



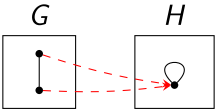
(c) A subgraph isomorphism.

Graph Morphisms

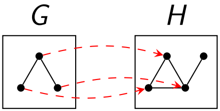
Def. graph isomorphism: a subgraph isomorphism which is a bijection of the vertices.

Unknown if in P or is NP-complete.

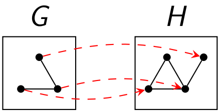
In $2^{O(\log^c n)}$ [Babai, 2016].



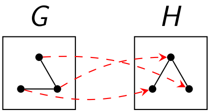
(a) A morphism.



(b) A monomorphism.



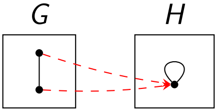
(c) A subgraph isomorphism.



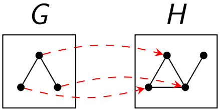
(d) An isomorphism.

Graph Morphisms

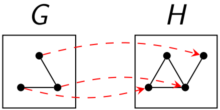
A pattern match: a monomorphism
Substructure search: monomorphism enumeration
Molecule equivalence: isomorphism detection



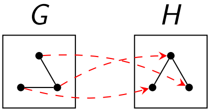
(a) A morphism.



(b) A monomorphism.

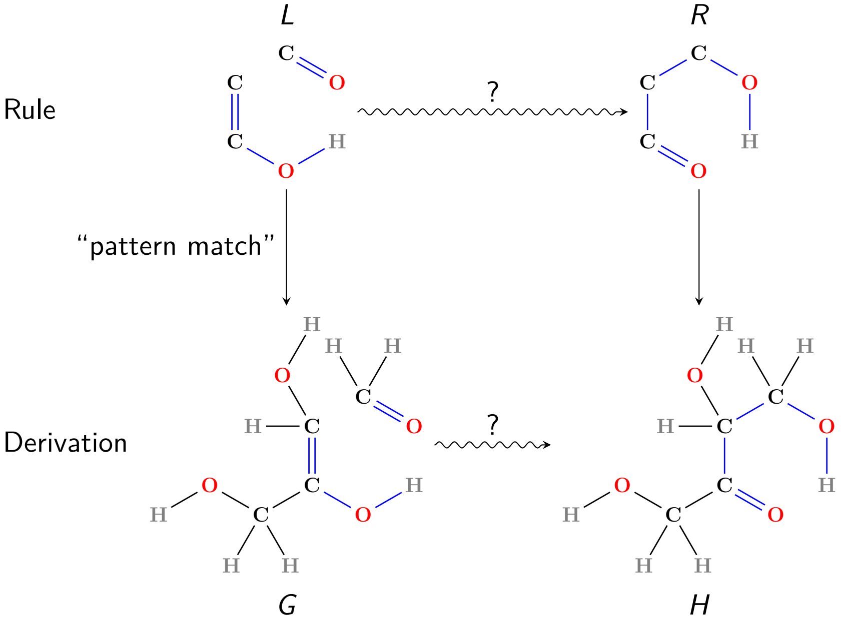


(c) A subgraph isomorphism.



(d) An isomorphism.

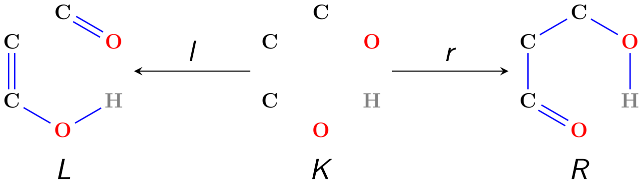
Graph Transformation



Graph Transformation Rules

Vertices and edges are either deleted, preserved, or added.

As a Double Pushout (DPO) rule $p = (L \overset{l}{\leftarrow} K \overset{r}{\rightarrow} R)$:

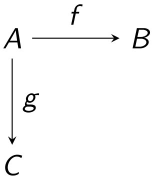


Intended semantics:

- $L \backslash K$ is deleted.
- K is preserved.
- $R \backslash K$ is added.
- For chemistry: l and r are monomorphisms.

Pushout

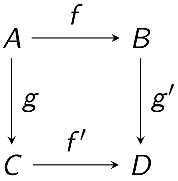
Given $C \leftarrow A \rightarrow B$,



Pushout

Given $C \leftarrow A \rightarrow B$, the pushout is f', g', D iff

- the square commutes: $fg' = gf'$, and



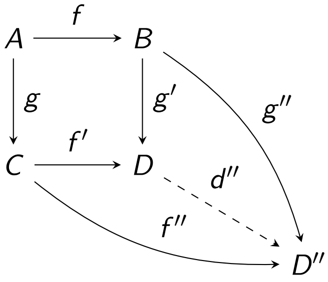
Pushout

Given $C \leftarrow A \rightarrow B$, the pushout is f', g', D iff

- the square commutes: $fg' = gf'$, and
- there are no "better" candidates:

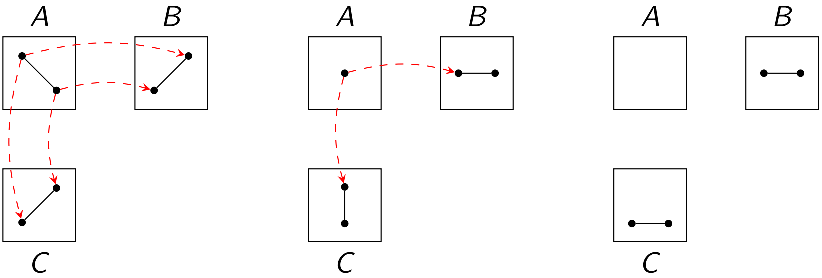
  for all commuting g'', f'', D'':

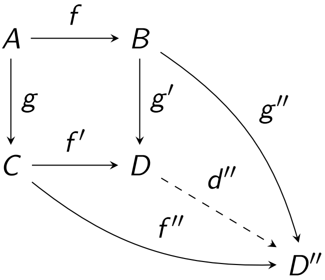
  d'' exists, commutes, and is unique.



Graph Pushouts (Generalised 'union' for Graphs)

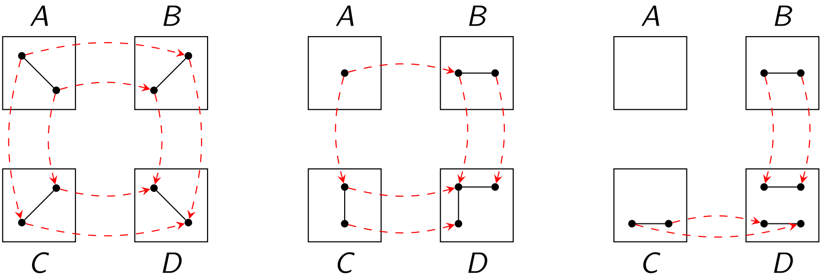
"The square must commute":

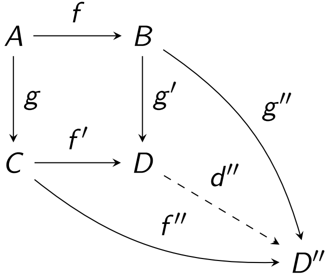




Graph Pushouts (Generalised 'union' for Graphs)

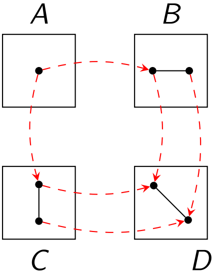
"The square must commute":





Graph Pushouts (Generalised 'union' for Graphs)

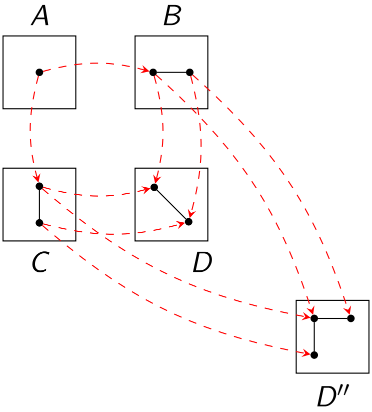
"There are no better candidates": counter-example



- It commutes!

Graph Pushouts (Generalised ‘union’ for Graphs)

“There are no better candidates”: counter-example

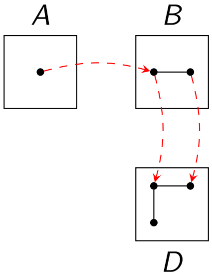


- It commutes!
- But D is “too small” (no commuting morphisms $D \to D''$).

Pushout Complement

Given $A \to B \to D$, find the pushout complement $A \to C \to D$:

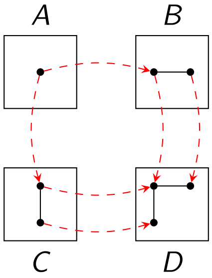
$B \to D \leftarrow C$ must be a pushout of $C \leftarrow A \to B$.



Pushout Complement

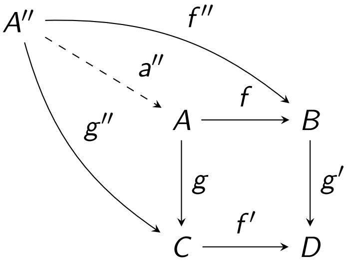
Given $A \to B \to D$, find the pushout complement $A \to C \to D$:

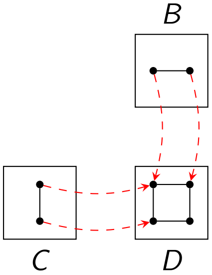
$B \to D \leftarrow C$ must be a pushout of $C \leftarrow A \to B$.



The Dual: Pullback

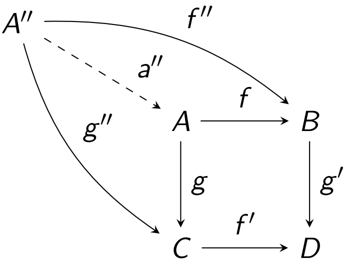
Given $C \to D \leftarrow B$, find the pullback $C \leftarrow A \to B$.

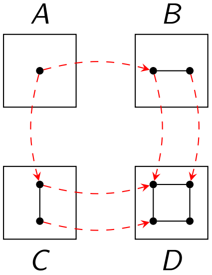




The Dual: Pullback

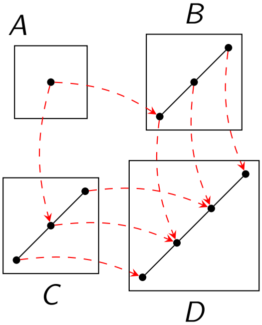
Given $C \to D \leftarrow B$, find the pullback $C \leftarrow A \to B$.





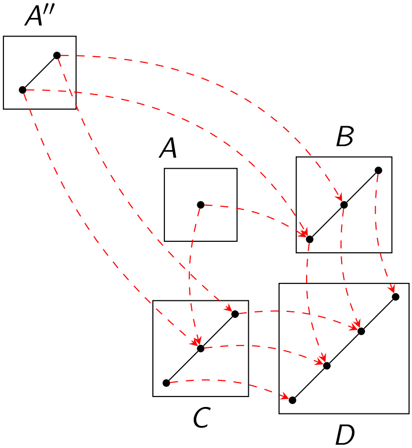
Graph Pullbacks

"There are no better candidates": counter-example

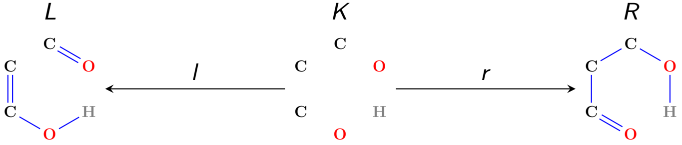


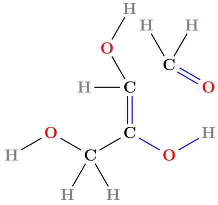
Graph Pullbacks

"There are no better candidates": counter-example



Rule Application

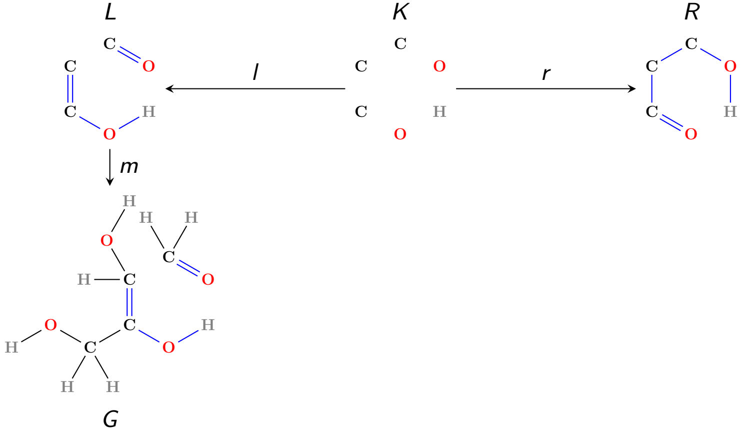




G

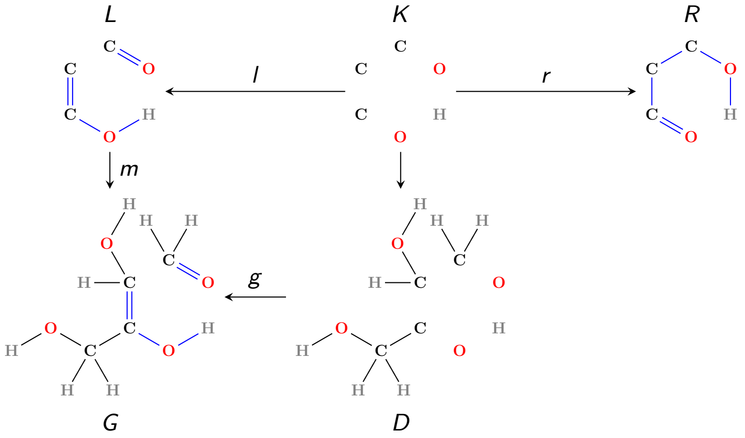
Given a rule $p = (L \xleftarrow{l} K \xrightarrow{r} R)$ and a graph G,

Rule Application



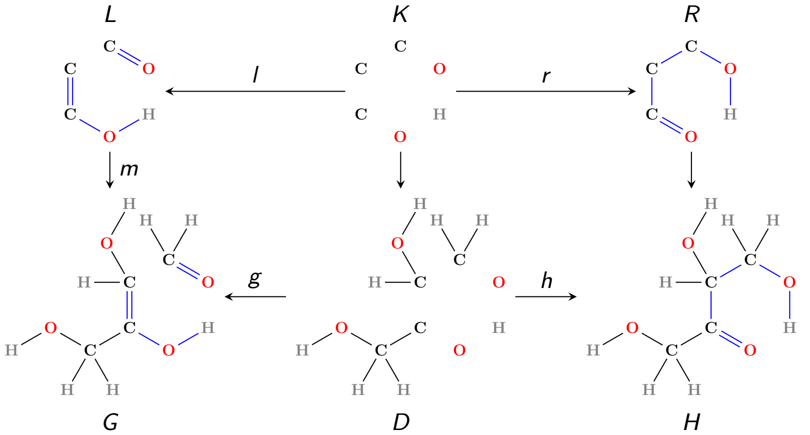
find a monomorphism $m\colon L \to G$,

Rule Application



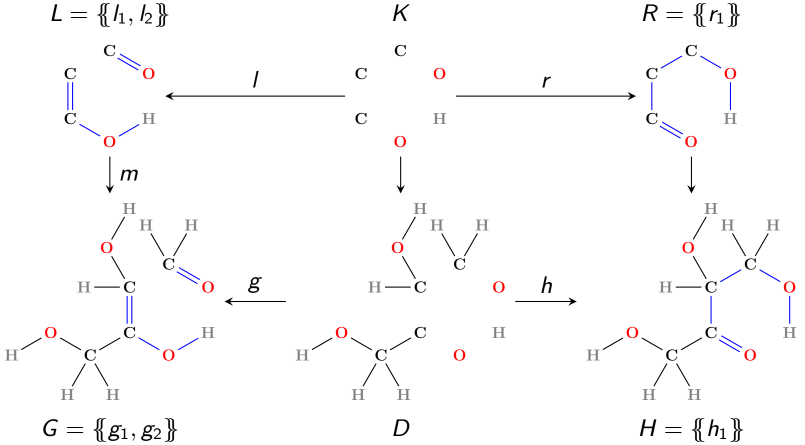
construct D as the pushout complement of $K \to L \to G$,

Rule Application



and construct H as the pushout object of $D \leftarrow K \rightarrow R$.

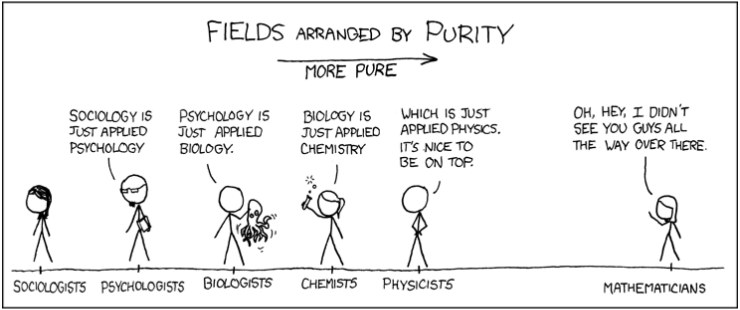
Chemical Rule Application



Two categories:

- For reactions: **C**, undirected graphs.
- For molecules: **C**′, connected undirected graphs.

Conclusions



Conclusions