

Chemical Transformation Motifs - Modelling Pathways as Integer Hyperflows

Jakob L. Anderson
Christopher Flamm
Daniel Merkle
Peter F. Stadler

SFI WORKING PAPER: 2016-04-006

SFI Working Papers contain accounts of scientific work of the author(s) and do not necessarily represent the views of the Santa Fe Institute. We accept papers intended for publication in peer-reviewed journals or proceedings volumes, but not papers that have already appeared in print. Except for papers by our external faculty, papers must be based on work done at SFI, inspired by an invited visit to or collaboration at SFI, or funded by an SFI grant.

©NOTICE: This working paper is included by permission of the contributing author(s) as a means to ensure timely distribution of the scholarly and technical work on a non-commercial basis. Copyright and all rights therein are maintained by the author(s). It is understood that all persons copying this information will adhere to the terms and constraints invoked by each author's copyright. These works may be reposted only with the explicit permission of the copyright holder.

www.santafe.edu



SANTA FE INSTITUTE

Chemical Transformation Motifs — Modelling Pathways as Integer Hyperflows

Jakob L. Andersen^{1,9}, Christoph Flamm^{2,8}, Daniel Merkle^{1,*}, Peter F. Stadler²⁻⁷

1 Department of Mathematics and Computer Science, University of Southern Denmark, Odense M DK-5230, Denmark

2 Institute for Theoretical Chemistry, University of Vienna, Wien A-1090, Austria

3 Bioinformatics Group, Department of Computer Science, and Interdisciplinary Center for Bioinformatics, University of Leipzig, Leipzig D-04107, Germany

4 Max Planck Institute for Mathematics in the Sciences, Leipzig D-04103, Germany

5 Fraunhofer Institute for Cell Therapy and Immunology, Leipzig D-04103, Germany

6 Center for non-coding RNA in Technology and Health, University of Copenhagen, Frederiksberg C DK-1870, Denmark

7 Santa Fe Institute, 1399 Hyde Park Rd, Santa Fe NM 87501, USA

8 Research Network Chemistry Meets Microbiology, University of Vienna, Wien A-1090, Austria

9 Earth-Life Science Institute, Tokyo Institute of Technology, Tokyo 152-8550, Japan

* E-mail: daniel@imada.sdu.dk

To the Memory of Harold J. Morowitz (1927-2016)

Abstract

We model chemical reaction networks as directed hypergraphs that are generated in rule-based manner, using graph grammars as models of given sets of reaction mechanisms. Graphs serve as abstractions of molecules. This provides a level of chemical realism sufficient to ensure conservation of mass, atom type, and charge. Atom maps, for instance, are thus consistently defined within the model. The generative approach pursued here goes beyond the necessarily static network models that need to be specified *a priori* and allows, in particular, application to network design problems.

Chemical pathways are represented by integer hyperflows. In contrast to more traditional approaches of flux balance analysis or elementary mode analysis we insist on integer-valued flows. Although this choice makes it necessary to solve possibly hard integer linear programs it conveys the advantage that more detailed mechanistic questions can be formulated and computed directly. Similarities and differences between our work and traditional approaches in metabolic network analysis are discussed in detail.

Three topics are used to demonstrate the applicability of the mathematical framework to real-life problems. We first explore the design space of possible non-oxidative glycolysis pathways and show that recent manual pathways can be further optimised. We then use a very general model of sugar chemistry to investigate the flows in the autocatalytic formose reaction and its relatives. Finally, we turn to the problem of recognizing complex autocatalytic cycles in large reaction networks, where we demonstrate how the TCA cycle and glyoxylate cycle as well as its combinations can be identified as autocatalytic.

Author Summary

Chemical reactions are typically presented as transformations of structural formulae. This leads to graph rewriting systems or graph grammars as the appropriate computational formalisation. On a more abstract level a reaction can be viewed as a single hyperedge, thus making a chemical reaction network or “universe” a directed hypergraph. Pathways then correspond to flows on these hypergraphs. We describe here a comprehensive theoretical framework and an integrated software system to model and analyse large-scale chemical networks. The combination of versatile strategies to generatively explore chemical reaction networks with very flexible ways to analyse networks in terms of integer network flows makes it possible to address novel types of questions. In particular we can systematically approach the problem of designing and optimising chemical reaction systems within a potentially infinite search space of alternative architectures.

1 Introduction

Large chemical reaction networks are at the basis of a wide variety of technical and scientific questions. Their scope ranges from models of metabolism in the context of both health and biotechnology, to complex processes in atmospheric and soil chemistry including the impact of pollution and strategies for environmental recovery, and to chemotechnical processes in fuel production and drug synthesis. The study of chemical reaction networks as entities in their own right, often with the aim of identifying some kind of pathways thus has a long-standing tradition. Well-established theories such as Flux Balance Analysis (FBA) [1–5], Elementary Flux Modes (EFM) [6–9], Extremal Pathways (ExPa) [10–12], or Chemical Organizations (CO) [13–15] have been developed for decades to analyse chemical reaction networks arising from a broad array of application scenarios. Each of these methods uses the stoichiometry of the networks, and hence the structure of the network as a directed hypergraph is respected. The approaches differ mostly in their aims. FBA is geared towards finding a single pathway that is optimal with respect to a predefined criteria, while both EFM and ExPa focus on fully characterising the space of potential pathways by enumerating certain corners of the underlying flux cone.

Chemical reaction networks are sometimes modelled as graphs rather than hypergraphs (e.g., see [16] for an overview). While this is sufficient for certain questions and has the advantage of admitting simple, computationally efficient algorithms, it disregards stoichiometric constraints and the structure and the many-to-many nature of the relationships implied by chemical reactions. Hence they are much less powerful with regard to integration of additional biological constraints [17].

In this contribution we present an intermediate model that respects the underlying stoichiometry of the network while providing sufficient flexibility to accommodate additional specialised constraints for the specific system under consideration. We demonstrate that with this model it is feasible in practice to perform targeted, large-scale enumeration of pathways according to a given optimisation criteria.

Much of the published literature views chemical reaction networks as *a priori* given graph or hypergraph structures. This point of view enables the algebraic methods of data analysis mentioned above, but at the same time disregards the inherently generative (constructive) nature of chemistry. Chemical reactions can produce new molecules that we may not even have thought of. In this contribution we also demonstrate that detailed network analysis and generative network construction cannot only be combined in a single framework, but also allows to formulate and answer novel types of questions.

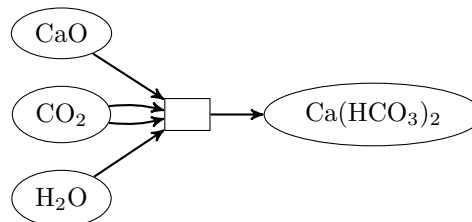


Figure 1. The reaction network with the reaction $\text{CaO} + 2 \text{CO}_2 + \text{H}_2\text{O} \longrightarrow \text{Ca}(\text{HCO}_3)_2$ as a directed multi-hypergraph. Each ellipse represent a vertex (molecule) in the hypergraph (reaction network), while a box represents a directed hyperedge (chemical reaction).

Chemical Networks and Fluxes The modelling approach used here consists of several interlinked components that are often subtly different from the models most commonly employed. First, our chemical reaction networks are directed multi-hypergraphs. This not only gives a well-defined “geometric” interpretation to the stoichiometric matrix but also naturally allows us to encapsulate catalysts. The simple reaction network with a single reaction $\text{CaO} + 2 \text{CO}_2 + \text{H}_2\text{O} \longrightarrow \text{Ca}(\text{HCO}_3)_2$, for example, has 4 vertices, representing the molecules CaO , CO_2 , H_2O , and $\text{Ca}(\text{HCO}_3)_2$. The network has a single hyperedge, whose tail is the educts CaO , H_2O , and two copies of CO_2 . The head of the hyperedge is the product $\text{Ca}(\text{HCO}_3)_2$. The network is visualised in Fig. 1.

Hypergraphs are a faithful representation of chemical reaction networks and in particular encode the complete stoichiometric information. The hypergraph representation can be extracted from published stoichiometric matrices, SBML files [18], pathway databases such as KEGG [19] or MetaCyc [20], or computed directly from generative models of chemistry [21–23].

In the hypergraph picture, chemical fluxes and pathways become *integer*-valued hyperflows [17, 24]. Hyperflows [25] are the natural generalization of flows on (directed) graphs to hypergraphs and thus provide a direct link to the rich computer science literature (see e.g., [26]). Flows on hypergraphs have been studied for restricted classes [27, 28] which however does not include chemical reaction networks. Indeed the problem of finding a maximum integer flow in a chemical reaction network (directed hypergraph) is NP-hard, even for bi-molecular reactions (bounded degree hyperedges) [29].

The insistence on integer rather than rational flows may sound unusual at first glance, since it forces us to deal with Integer Linear Programs (ILP) instead of the Linear Programs arising from FBA, EFM, or ExPa problems, which have polynomial time solutions. In contrast, ILP is known to be NP-hard in the general case [30]. This is a price well worth paying as we shall see: First, some questions – such as the identification of autocatalytic subsystems – can be asked in the traditional framework only with the help of additional constraints and conditions that augment the FBA framework. Second, we will show that there are chemically realistic cases in which the LP and ILP solutions are vastly different. Since chemical reactions transform individual molecules, ILP solutions admit a direct mechanistic interpretation, while non-integer LP solutions may be superpositions of distinct mechanisms. Of course, many such issues have been observed and discussed in the framework of FBA, EFM, or ExPA, where they can be dealt with by post-processing the results. The advantage of the ILP framework is that it directly yields more interpretable answers also in difficult cases. Given the efficiency of modern ILP solvers, we find the computational price is worth paying. ILP also offers additional benefits. For example, it becomes straightforward to sample or

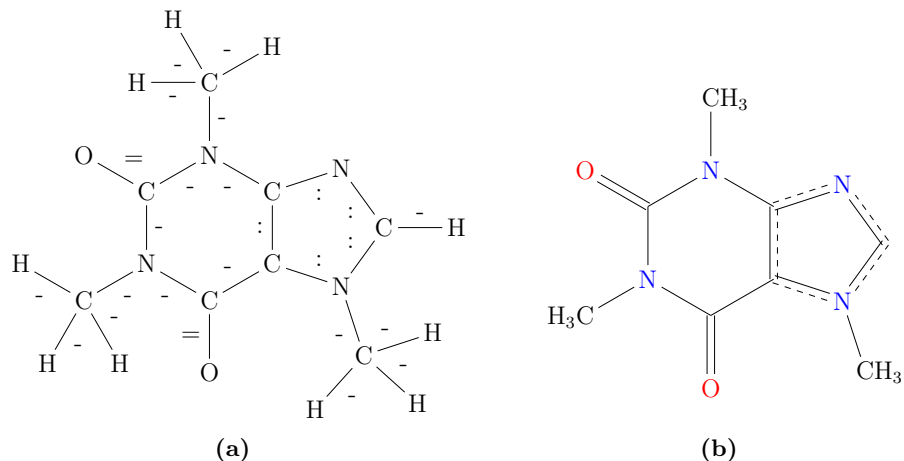


Figure 2. A graph representing the molecule caffeine visualised as (a) the raw labelled graph and (b) in an abbreviated style familiar to chemists. Aromaticity is depicted with special bonds, as opposed to using alternating single and double bonds, to properly show the actual molecule encoding.

exhaustively enumerate suboptimal solutions in a principled manner. Furthermore, additional ILP constraints have been employed to prune trivial pathways involving reversible reactions [17, 24].

Graph Grammars as Models of Chemistries The second key component of our approach is the generation of the reaction networks themselves by means of graph transformation systems. This forces us to have an explicit representation of a molecule as a graph rather than to treat molecules as mere labels for the nodes of the reaction hypergraph. Adopting the ideas of chemical graph theory [31–33], our molecules are connected labelled graphs. Vertex labels encode atom type and charges, while edge labels indicate bonds types. Since double and triple bonds are regarded as labels, molecule graphs are simple, i.e., they do not contain loops or parallel edges, see Fig. 2 for an illustration.

When molecules are represented as graphs it becomes natural to regard chemical reactions as graph transformations [34–39]. Several formal frameworks have been described in the literature [40]. We find the Double Pushout (DPO) formalism particularly intuitive for modelling chemistry [22]. A DPO transformation rule encodes a *reaction pattern*, i.e., a generalised mechanism that can be applied on many different sets of molecules. Formally, a rule $p = (L \xleftarrow{l} K \xrightarrow{r} R)$ consist of three graphs (fragments of molecules) L , R , and K called the left, right, and context graph, respectively. In addition, the rule specifies two graph morphisms l and r that determine how the context K is embedded in the left and right graph.

A rule is applied to a graph G if a matching morphism $m: L \rightarrow G$ exists that fulfil an additional consistency criterion. The chemical intuition is that L is a molecule fragment that must be present in the educt molecules G for the reaction to take place. From p , G , and m a new graph H can be derived that can be interpreted as a modified version of G with the copy of L replaced by R . Thus H represents the product molecules. We denote such a derivation (chemical reaction) as $G \xrightarrow{p,m} H$ and visualise it

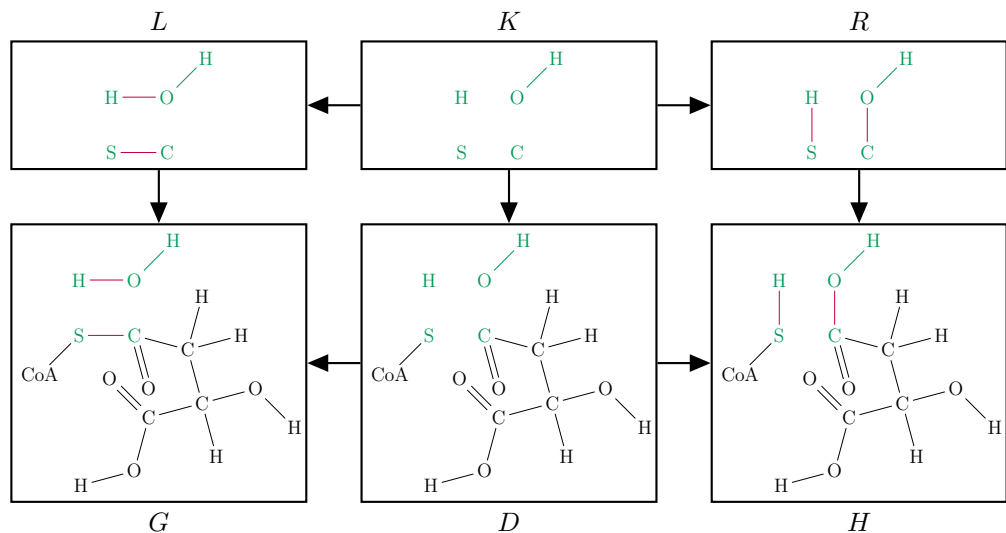


Figure 3. Abstract derivation $G \xrightarrow{p,m} H$ with $p = (L \xleftarrow{l} K \xrightarrow{r} R)$. A concrete chemical derivation (reaction), though with a large part of Coenzyme A abstracted into a single vertex. Note that all hydrogen atoms are shown explicitly and that G as well as H consist of two components. Edges (bonds) being changed during the transformation are shown in red. The context K of the rule, and its matches in the other graphs, is shown in green.

as the commutative diagram

$$\begin{array}{ccccc}
 L & \xleftarrow{l} & K & \xrightarrow{r} & R \\
 m \downarrow & & d \downarrow & & n \downarrow \\
 G & \xleftarrow{m} & D & \xrightarrow{n} & H
 \end{array} \tag{1}$$

This notation has a precise mathematical meaning within category theory and forms the basis for the well-definedness of the machinery of graph rewriting. It can intuitively be understood in terms of molecules that react and the parts of the molecules that define the reaction mechanism, i.e., the rule itself, as in Fig. 3.

Chemical transformation rules have a very special structure within the realm of graph rewriting operations because atoms are preserved in the course of a reaction. Only the chemical bonds are rearranged. Thus all morphisms are necessarily injective and $L \setminus K$ and $R \setminus K$ contains only edges. Each derivation therefore implies a well-defined atom-atom map between educts and products. A concrete example of a chemical derivation is shown in Fig. 3.

Strategies to Explore a Chemical Universe A chemical universe [41], specified by a collection of reaction rules (i.e., particular types of chemical reactions of interest and a collection of start molecules) comprises all molecules that can be generated from the starting molecules by repeated application of the available rules. Each graph (molecule) thus becomes a vertex in the hypergraph and each derivation $G \Rightarrow H$ becomes a directed hyperedge.

It is important to note that chemical universes can be very large. Polymerization reactions, for example, generate mathematically infinite chemical universes. Brute force enumeration is therefore usually infeasible. Well-defined *exploration strategies* [23] can

be used instead that apply graph transformation rules in a guided manner and provides preference, pruning, and filtering rules for molecules that implement a wide array of constraints. A directed hypergraph (chemical reaction network) is obtained by recording the direct derivations discovered during the execution of a strategy.

Application Scenarios We demonstrate the usefulness of combining a generative for chemical reaction networks with the analysis of the networks in terms of integer hyperflows in the context of several different applications. The question to what extent and in what sense biochemical pathways are optimal dates back to the late 1980s. Among the first studies of this type was Meléndez-Hevia’s analysis of sugar rearrangements in the pentose-phosphate pathway (PPP) [42]. The PPP is used in cells to transform hexoses (C_6 sugars e.g. glucose a prominent carbon sources) into pentoses (C_5 sugars, integral components of nucleic acids such as RNA and DNA). The conversion of five hexoses to six pentoses can be achieved in many different ways using only established enzyme reaction mechanisms [43] such as transketolase and transaldolase (which transfer C_2 - or C_3 -fragments between sugars). It turned out that nature implements this conversion with the least number of steps.

Following this line of reasoning we enumerate in section 5.1 the possible non-oxidative glycolysis pathways. Here, the design space is much larger than what one can hope to cover manually or by *ad hoc* approaches. We automatically generate and compare the alternatives to the single pathway “designed by intuition” in [44] and show that superior solutions exist. Microbial synthesis of plant-derived secondary metabolites currently is a hot topic in metabolic engineering [45, 46]. Impressive examples include the synthesis of the anti-malarial drug precursor artemisinic acid [47], and of (S)-reticuline [48], an important intermediate towards the benzyloisoquinoline alkaloids codein or morphine. Synthetic Biology holds promise to transform the entire process of discovering and manufacturing pharmaceuticals. The most common strategy is to transplant an existing biosynthetic pathway, e.g. from a plant, into the host microbe. For many natural products of pharmaceutical interest, however, the biosynthetic pathway is only partially known, or complete unknown. In such cases computational strategies for *de novo* rational design, such as ours, are necessary, that identify higher-order functional transformation networks with predictable behaviour from existing enzyme reactions.

We then proceed to a detailed analysis of the flows in the autocatalytic formose reaction in a general sugar chemistry network. We ask how autocatalytic cycles are related to each other in functional cascades and how these intricate structures “communicate” to the embedding background network of potentially “destructive” side reactions. The ability to answer this question is essential for the characterization and classification of large-scale chemical reaction networks. Networks involving distributed autocatalysis and their properties are the core objects to understand the transition from prebiotic chemical processes to the emergence of life. While quantum chemical methods are efficient to predict the structure and reactivity of small molecules [49], techniques, such as the graph grammar approach, located at a more coarse-grained level of theory, are necessary to analyse the topological and functional organisation of complex chemical reaction networks. Furthermore, higher order properties such as robustness or sensitivity to perturbations can only be attacked within this broader systems perspective.

Finally, we turn to the problem of recognizing complex autocatalytic cycles in large reaction networks. Using a variety of different objective functions we show that it is possible to recognise the TCA cycle, the glyoxylate cycle, as well as an autocatalytic combination of the two cycles without prior knowledge. Upon inclusion of parts of the pyruvate conversion pathway, furthermore, the reverse TCA cycle also becomes autocatalytic. In prebiotic chemistry [50], autocatalysis has long been hypothesized to

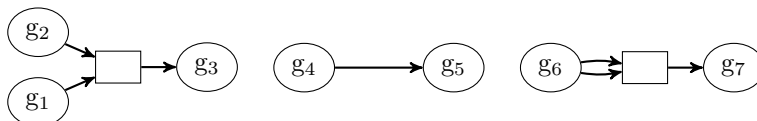


Figure 4. Visualization of an abstract reaction network consisting of the reactions $g_1 + g_2 \longrightarrow g_3$, $g_4 \longrightarrow g_5$, and $2g_6 \longrightarrow g_7$.

be one of the dominating mechanisms for focusing distributed mass in a combinatorial mixture of reversibly interconverting molecules towards a small set of molecules, which could have paving the way for the emergence of self-sustaining non-enzyme catalysed primitive metabolic pathways. One of the main problems here, however is that only a handful of prebiotically plausible autocatalytic reaction cycles have been described. Furthermore other forms of global autocatalytic behaviour derived from cross-catalysis or cascade catalytic mechanisms are conceivable. Therefore, a computational approach, which allows to formally specify such autocatalytic behaviour, and subsequently enables their efficient identification in arbitrary chemical reaction networks, makes a valuable contribution to the origin of life research. Once novel instances are found, expensive experimental and first-principle computational methods can be applied to study their dynamic behaviour.

2 Model Description

In this section we develop a formal model for pathways in reaction networks. For conciseness, we entirely use the language of hypergraphs in this section.

2.1 Directed Multi-Hypergraphs

A directed multi-hypergraph $\mathcal{H} = (V, E)$ is an ordered pair of a vertex set V and a set of directed hyperedges E . Each edge $e \in E$ is itself a pair (e^+, e^-) of multisets (hence “multi-”hypergraph) of vertices $e^+ \subseteq V$ and $e^- \subseteq V$, often called the tail and head of e . In the following we will refer to directed multi-hypergraphs simply as hypergraphs. Fig. 4 gives an example of the general visualisation scheme we use throughout this contribution.

Since we will need to use both normal sets and multisets we will introduce the following notation for multisets. When constructing a multiset we use double curly brackets (i.e., $\{\{\dots\}\}$) to distinguish them from normal set constructors, $\{\dots\}$. For a multiset Q and some element q we use $m_q(Q)$ to denote the number of occurrences of q in Q . We introduce a multi-membership operator, \in_m , used in iteration contexts. E.g., $\sum_{q \in_m Q} 1 = m_q(Q) = |\{\{q \in_m Q\}\}|$.

We first describe the basic pathway model and a simple notion of catalysis and autocatalysis. This model allows for misleading, or chemically “uninteresting”, pathways. An expanded model is therefore described to narrow the space of pathways.

Note that the core model aims at formalising the notion of a chemical pathway, and does as such not include any optimality criteria. However, the model forms the basis of a practical implementation in terms of integer linear programs (ILP), which we will describe in detail in the following section. It is trivial, therefore, to add a linear objective function.

2.2 Integer Flows on Extended Hypergraphs

Throughout, we assume that $\mathcal{H} = (V, E)$ is a directed multi-hypergraph. Syntactically we will use a superscripted plus $^+$ to refer to “out”-related elements relative to vertices

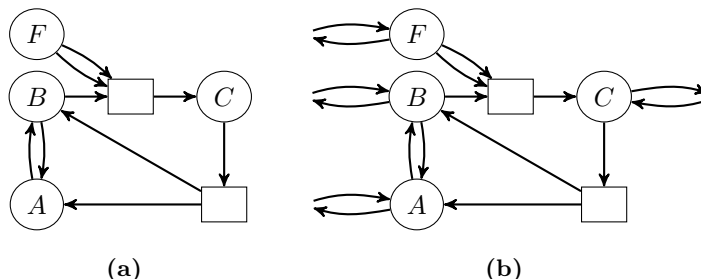


Figure 5. (a) a small network, \mathcal{H} . (b) the extended network, $\overline{\mathcal{H}}$. Note that most of the input/output edges in the extended network will be constrained in the final formulation, and thus for many chemical networks many of these edges will effectively be removed to model specific interface conditions.

(e.g., out-edges), and a superscripted minus $-$ to refer to “in”-related elements.

We will need a mechanism to introduce and extract molecules from the network, and we therefore define the *extended hypergraph* $\overline{\mathcal{H}}$ of \mathcal{H} as

$$\begin{aligned}\overline{\mathcal{H}} &= (V, \overline{E}) \\ \overline{E} &= E \cup E^- \cup E^+ \\ E^- &= \{e_v^- = (\emptyset, \{v\}) \mid v \in V\} \\ E^+ &= \{e_v^+ = (\{v\}, \emptyset) \mid v \in V\}\end{aligned}\tag{2}$$

which has additional “half-edges” e_v^- and e_v^+ , for each $v \in V$. These explicitly represent potential input and output channels to and from \mathcal{H} , see Fig. 5.

Network flows on graphs is a well-studied topic and can be used as a intuitive model for many problems [26, 51]. Hyperflows are the natural generalization of network flow models from graphs to hypergraphs. Flows on several restricted types of hypergraphs have been considered in the literature, in particular so-called gain-free hypergraphs [52] and hypergraphs with single-vertex heads, i.e., $|e^-| = 1$ for all edges [27, 28]. Hyperflows are very similar to the notion of chemical fluxes used in, e.g., Flux Balance Analysis [5], but as a core difference we will here restrict the model to *integer* hyperflows. In section 3 we provide a detailed comparison to Flux Balance Analysis and the consequences of the integrality constraint. Integer coefficients are also employed in [24], where the term *tick vector* is used for what call here a pathway.

Recall the multiplicity function for multisets, where we for a vertex $v \in V$ and an edge $e \in \overline{E}$ can write $m_v(e^-)$ for the number of occurrences of v in the head of e (and $m_v(e^+)$ for the tail). We use $\delta^+(\cdot)$ and $\delta^-(\cdot)$ to denote a set of incident out-edges and in-edges respectively, and thus use $\delta_{\overline{E}}^+(v)$ as the set of out-edges from v , restricted to the edge set \overline{E} , i.e., $\delta_{\overline{E}}^+(v) = \{e \in \overline{E} \mid v \in e^+\}$. Likewise, $\delta_{\overline{E}}^-(v)$ denotes the restricted set of incident in-edges of v .

Definition. An integer hyperflow on $\overline{\mathcal{H}}$ is a function $f: \overline{E} \rightarrow \mathbb{N}_0$ satisfying, for each $v \in V$ the conservation constraint

$$\sum_{e \in \delta_{\overline{E}}^+(v)} m_v(e^+)f(e) - \sum_{e \in \delta_{\overline{E}}^-(v)} m_v(e^-)f(e) = 0\tag{3}$$

We mostly speak of integer hyperflows, and will for brevity refer to them simply as flows.

In order to constrain the in- and out-flow to certain vertices we specify set of inputs (sources) $S \subseteq V$ and outputs (targets/sinks) $T \subseteq V$. Thus

$$f(e_v^-) = 0 \quad \forall v \notin S \quad \text{and} \quad f(e_v^+) = 0 \quad \forall v \notin T\tag{4}$$

serve as additional constraints in an I/O-constrained extended hypergraph, which is completely specified by the triple (\mathcal{H}, S, T) .

We adopt the notion of an *overall flow* from the chemical *overall reaction* for a pathway, which is simply a convenient notation for the I/O flow. For a flow f we syntactically write the overall flow as

$$f(e_{v_1}^-) v_1 + \cdots + f(e_{v_{|V|}}^-) v_{|V|} \longrightarrow f(e_{v_1}^+) v_1 + \cdots + f(e_{v_{|V|}}^+) v_{|V|}$$

However, as usual for chemistry we omit the terms with vanishing coefficients.

Flows are non-negative by definition. While we for reversible reactions could have allowed negative flows (see Sec. 3), we adhere to the usual framework of flow problems. It is therefore necessary to model every reversible reaction by two separate edges $e = (e^+, e^-)$ and $e' = (e^-, e^+)$. This separation of the flow will later allow us to define useful chemical constraints on the flow.

A capacity function $u: \bar{E} \rightarrow \mathbb{N}_0$, finally, limits the flow from above, i.e., $f(e) \leq u(e)$, as in many typical flow problems. We will not make use of the capacity function in this contribution.

2.3 Specialised Flows – Overall (Auto)catalysis

The nature of autocatalysis is that, the product of a chemical reaction catalyses its own formation. This interaction leads to an exponential time behaviour in the growth characteristics of the product, as well as, to a positive correlation of initial concentration and the reaction rate. Autocatalytic reactions have been investigated for over a century and instances of this behaviour have been found in a wide range of topologically different chemical systems, which are based on a rich chemical setup (for a recent review see [53]). Autocatalysis plays a major role in the processes of life. Usually several reactions are organised in a cyclic network to achieve the proper topology for autocatalysis. Clearly the reaction flux around the cycle must be significantly larger than the flux along potential draining reactions, which reroute flux away from the cycle, to achieve turnover. If autocatalytic cycles are coupled together in a cyclic, mutually autocatalytic structure, a new level of organization, called hypercycle, is reached. The concept of a hypercycle, as a principle of natural self-organization, was proposed by Schuster and Eigen [54–56].

We here define a simple notion of both catalysis and autocatalysis in terms of the I/O flow of a network. As we only constrain the flow of the overall reaction, we call this *overall (auto)catalysis*. Catalysis means in chemical terms that a molecule, the catalyst, is first consumed by some reaction and then regenerated by a subsequent reaction in such a way that overall the catalyst is neither consumed nor produced. We therefore say that a vertex $v \in V$ is *overall catalytic* in a flow f if and only if (i) the input and output flows of v are non-zero and (ii) the input and output flows of v are equal, i.e., iff

$$0 < f(e_v^-) = f(e_v^+) \tag{5}$$

Similarly, autocatalysis, refers to a situation where a molecule is consumed in a reaction only to be (re)produced in subsequent reactions in higher quantities than what has been consumed originally. In terms of flow we say a vertex $v \in V$ is *overall autocatalytic* in a flow f if and only if

$$0 < f(e_v^-) < f(e_v^+) \tag{6}$$

We extend the terminology to say that a flow f is overall (auto)catalytic if some vertex is overall (auto)catalytic in f .

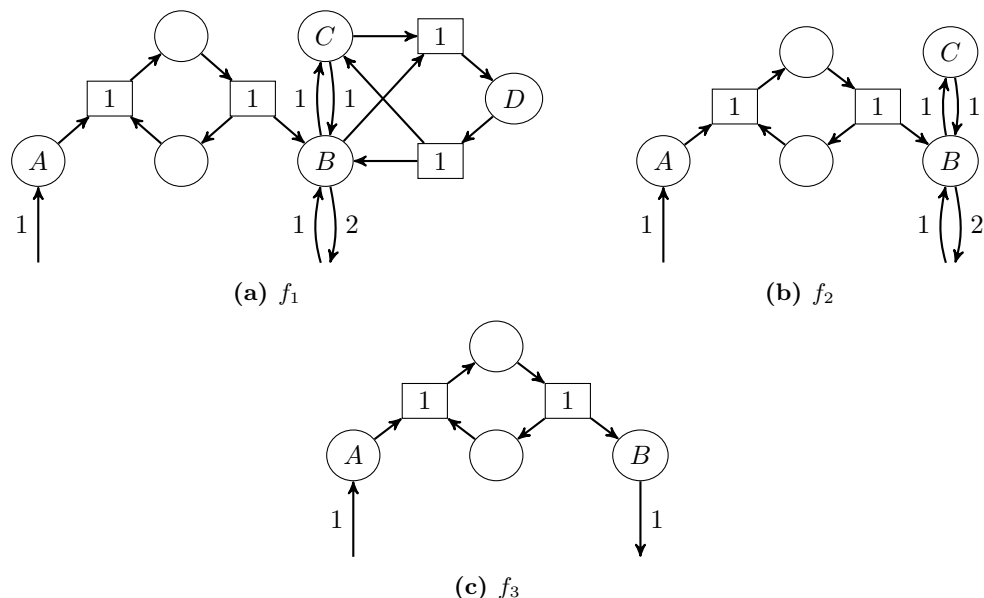


Figure 6. Simplification of a flow f_1 to an equivalent flow f_3 , by removal of futile 2-edge sub-pathways. (a) The molecule D is created through $B + C \longrightarrow D$, but can only be interpreted as being consumed through the reverse reaction. (b) After removal of 1 flow from the reactions $B + C \longleftrightarrow D$, the molecule C now participates in a futile 2-edge flow. (c) Removing 1 flow from $B \longleftrightarrow C$ and the I/O edges $\emptyset \longleftrightarrow B$, we arrive at the simplest flow.

2.4 Chemically Simple Flow and Vertex Expansion

Modelling reversible reactions as pairs of irreversible ones gives rise to pathways where both an edge e and its inverse e^{-1} have positive flow. Consider the hypergraph annotated with a flow in Fig. 6a. Here we see three pairs of reversible reactions with positive flow: $B + C \longleftrightarrow D$, $B \longleftrightarrow C$, and the I/O reactions $\emptyset \longleftrightarrow B$. However, we can argue that this flow is not “simple” in the sense that there is no interpretation of the flow without a futile conversion of matter. This problem has of course been recognized in the literature. Additional ILP constraints were used in [17, 24] to prune such cases. However, this approach seems quite difficult to control in practise. Disallowing all 2-cycles as in [24], for example, turns out to be too restrictive. Therefore we advocate a refinement of the network model itself.

In the pathway a single copy of D is created, through the reaction $B + C \longrightarrow D$, and it can only be routed into a single reaction, $D \longrightarrow B + C$. The sub-pathway $B + C \longrightarrow D \longrightarrow B + C$ is thus a futile 2-edge branch that we can simplify away, yielding the equivalent flow in Fig. 6b. The same reasoning can now be applied to C , and subsequently B , resulting in the flow depicted in Fig. 6c.

Formally we say that a flow f is not *chemically simple* if there is a vertex $v \in V$ that has only one in-edge $e \in E$ with positive flow and only one out-edge $e' \in E$ with positive flow, where the two edges are each others inverse, $e' = e^{-1}$.

The original flow in Fig. 6a fulfils the requirement for overall autocatalysis in vertex B (Eq. (6)), but clearly the in-flow of 1 B is not involved in the extra production of B , which goes against the idea of general autocatalysis. The simplified flow, Fig. 6c, is however not overall autocatalytic, and it is therefore desirable to constrain the model such that non-simple flows are not possible, in order to further approach a precise characterisation of autocatalysis.

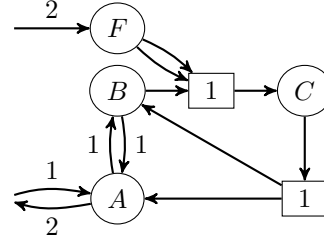


Figure 7. Example of a flow with meaningful 2-cycles, in the network from Fig. 5b. Only edges with non-zero flow are shown.

From the shown example it is tempting to simply disallow all 2-cycles of flow. This is the approach effectively used in FBA-related methods (see Sec. 3), and also in flows on normal graphs [26, 51]. However, as illustrated in Fig. 7, this is too strong a constraint. We can interpret this flow such that no flow is directly reversed:

- | | |
|--|--|
| 1. $\emptyset \longrightarrow A$ | 5. $C \longrightarrow A + B$ |
| 2. $A \longrightarrow B$ | 6. $B \longrightarrow A$ |
| 3. $\emptyset \longrightarrow F$ twice | 7. $A \longrightarrow \emptyset$ twice |
| 4. $B + 2F \longrightarrow C$ | |

Since this interpretation is a series of chemically meaningful transformations, it should not be excluded from the pathway model.

To facilitate the constraints that disallow flows that are not chemical simple we expand the extended hypergraph into a larger network. Each vertex is expanded into a subnetwork that represents the routing of flow internally in the expanded vertex. Formally, for each $v \in V$

$$V_v^- = \{u_{v,e}^- \mid \forall e \in \delta_E^-(v)\} \quad (7)$$

$$V_v^+ = \{u_{v,e}^+ \mid \forall e \in \delta_E^+(v)\} \quad (8)$$

$$E_v = \{(\{u^-\}, \{u^+\}) \mid u^- \in V_v^-, u^+ \in V_v^+\}$$

That is, v is replaced with a bipartite graph $(V_v^- \cup V_v^+, E_v)$ with the vertex partitions representing the in-edges and out-edges of v respectively, and the edge set being the complete set of edges from the in-partition to the out-partition. We say that E_v is the set of *transit edges* of v .

The original edges are reconnected in the natural way; for each $e = (e^+, e^-) \in \bar{E}$ the reconnected edge is \tilde{e} :

$$\begin{aligned} \tilde{e} &= (\tilde{e}^+, \tilde{e}^-) \\ \tilde{e}^- &= \{\{u_{v,e}^- \mid v \in_m e^-\}\} \\ \tilde{e}^+ &= \{\{u_{v,e}^+ \mid v \in_m e^+\}\} \end{aligned}$$

We finally define the expanded hypergraph as

$$\begin{aligned} \tilde{\mathcal{H}} &= (\tilde{V}, \tilde{E}) \\ \tilde{V} &= \bigcup_{v \in V} V_v^- \cup \bigcup_{v \in V} V_v^+ \\ \tilde{E} &= \bigcup_{v \in V} E_v \cup \{\tilde{e} \mid e \in \bar{E}\} \end{aligned}$$

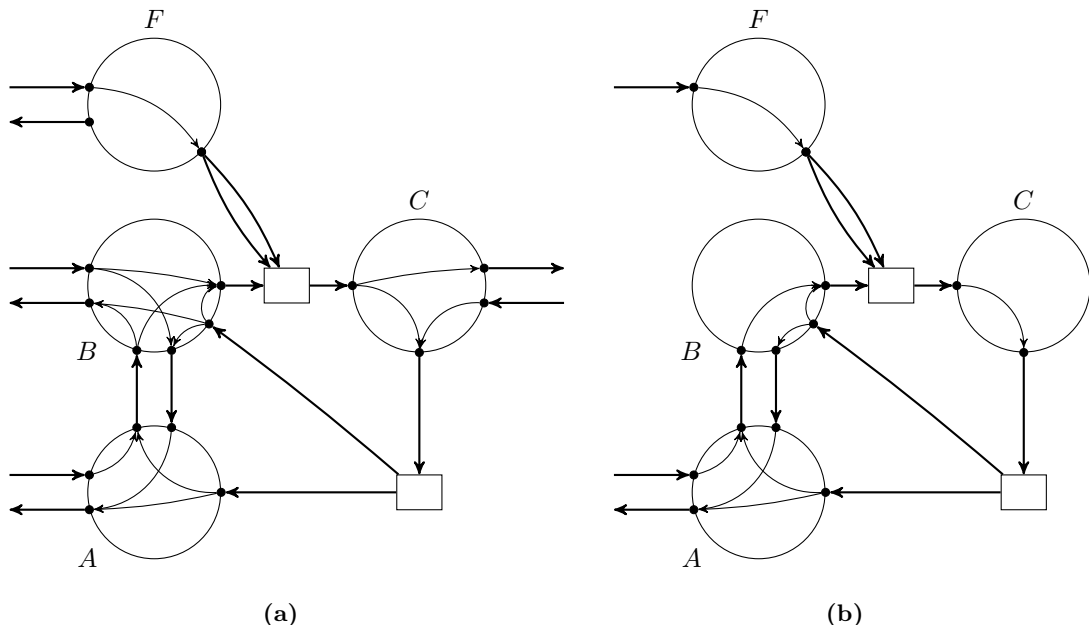


Figure 8. The network from Fig. 5b expanded into $\tilde{\mathcal{H}}$. The vertices of $\tilde{\mathcal{H}}$ are the small filled circles, while the large circles, A , B , C and F , only serves as visual grouping of the actual vertices. (a) the expanded network without transit edges constrained in Eq. (10). (b) the expanded network, simplified using the source set $S = \{A, F\}$ and sink set $T = \{A\}$.

We expand the definition of a flow function to $f: \tilde{E} \rightarrow \mathbb{N}_0$ and pose the usual conservation constraints, but on $\tilde{\mathcal{H}}$: for all $v \in \tilde{V}$

$$\sum_{e \in \delta_{\tilde{E}}^+(v)} m_v(e^+)f(e) - \sum_{e \in \delta_{\tilde{E}}^-(v)} m_v(e^-)f(e) = 0 \quad (9)$$

The I/O constraints translates directly to the expanded network. In Sec. 2.5 we formally describe the relationship between flows on the extended and the expanded network.

Using the expanded network we can prevent flow from being directly reversed; a flow f must satisfy that for every pair of mutually inverse edges $e = (e^+, e^-), e' = (e^-, e^+) \in \tilde{E}$, we have

$$f((u_{v,e}^-, u_{v,e'}^+)) = 0 \quad \forall v \in \tilde{E} \quad (10)$$

Fig. 8 shows the expanded version of the network from Fig. 5b, with these constraints in effect.

Note that the expansion of the network also opens the possibility of forbidding other 2-sequences of edges, and in general the possibility of posing constraints on the routing of flow internally in vertices.

When querying for chemical pathways with partially unknown I/O specification we have found it useful to distinguish between reversible reactions that are in the original network \mathcal{H} and the reversible I/O reactions. That is, we may choose to not pose the above constraints on transit edges $(u_{v,e}^-, u_{v,e'}^+)$ when $e = e_v^-$ and $e' = e_v^+$, thus allowing excess input-flow to be routed directly out of the network again.

Fig. 7 showed a valid flow with a meaningful 2-cycle. The expanded flow is shown in Fig. 9, and we note that no 2-cycles exist in this flow.

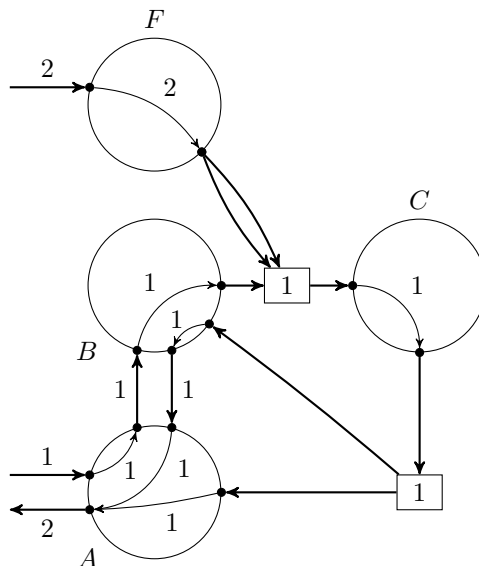


Figure 9. The example flow from Fig. 7 in the expanded network, with only edges with non-zero flow shown. Note that no 2-cycles exist in this flow.

Overall Catalysis and Autocatalysis In Eq. (5) and (6) we defined the I/O constraints for overall catalysis and autocatalysis. These constraints are converted in the obvious manner to the expanded network $\tilde{\mathcal{H}}$. However, the expanded network reveals another possibility for somewhat misleading flows, exemplified in Fig. 10a. Vertex A is overall autocatalytic, but is also utilised as an intermediary molecule. The same autocatalytic motif can be expressed by a simpler flow, Fig. 10b.

In the interest of finding the simplest (auto)catalytic flows, we introduce the following constraints. Let f be a flow and $v \in V$ a vertex satisfying the I/O constraints for overall catalysis (5) (resp. overall autocatalysis (6)). In the expanded network f must additionally satisfy the transit constraints (note that $\delta_E^\pm(v)$ does not include the I/O edges):

$$f((u_{v,e'}^+, u_{v,e''}^-)) = 0 \quad \forall e' \in \delta_E^-(v), e'' \in \delta_E^+(v)$$

That is, all transit flow in an overall (auto)catalytic vertex must flow either from the input edge e_v^- or towards the output edge e_v^+ .

Exclusive Autocatalysis If a compound is overall autocatalytic, it merely means that; if it is available then even more can be produced. However, this does not mean that it cannot be produced solely by the other input compounds. Solutions can therefore be found that may be surprising. One such solution is illustrated in Fig. 11. As a variant of our definition of autocatalysis we define that a vertex v , is *exclusively overall autocatalytic* if and only if it is overall autocatalytic and is not *trivially reachable* from the other input vertices, $S \setminus \{v\}$. A vertex v is trivially reachable from a vertex set S' if it can be marked during a simple breadth-first marking of the hypergraph $\mathcal{H} = (V, E)$. For completeness, the pseudocode is shown in Algorithm 1.

Note that breadth-first marking of hypergraphs, and variations thereof, has in the literature also been referred to as finding *scopes* of molecules [57, 58]. Breadth-first marking has in those studies been used alone to analyse metabolic networks, and define set-theoretical notions of pathways and later of autocatalysis [59]. The methods thus do

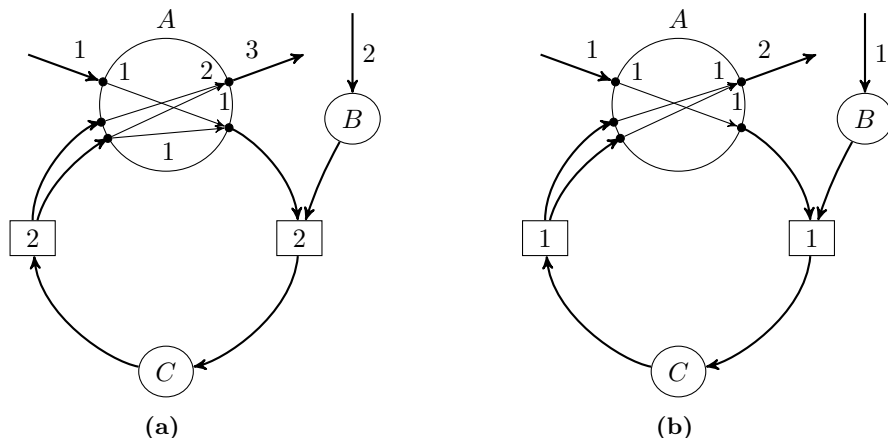


Figure 10. A simplified network with an overall autocatalytic flow. (a) the vertex A is both overall autocatalytic and an intermediate vertex in the flow. (b) the same motif for overall autocatalysis, but without A being an intermediate vertex.

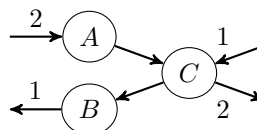


Figure 11. The vertex C is overall autocatalytic, but not autocatalytic in the chemical sense.

not have focus on the underlying mechanism of the pathways, which is our aim in this contribution.

2.5 Properties of the Expanded Hypergraph

The expansion of the networks obviously changes the size of the underlying model, and it is therefore necessary to investigate how large the expanded network can get, in order to bound the complexity of algorithms.

Size of the Expanded Network For a directed multi-hypergraph $\mathcal{H} = (V, E)$, let the size of it be denoted by $\text{size}(\mathcal{H}) = |V| + |E| + \sum_{e \in E} (|e^+| + |e^-|)$. Note that if the hypergraph is seen as a bipartite normal graph, then this size corresponds to the number of vertices and edges.

Algorithm 1: Breadth-first marking of a hypergraph, from a set of input vertices.

Input : A directed (multi-)hypergraph $\mathcal{H} = (V, E)$.

Input : A set of starting vertices $S' \subseteq V$.

Output : A marked subset of the vertices.

foreach $v \in S'$ **do** mark v

while no more hyperedges can be marked **do**

foreach $(e^+, e^-) \in E$ **do**

if all $v \in e^+$ are marked **then**

 mark e

foreach $v \in e^-$ **do** mark v

Proposition. *The size of the extended network and the expanded network is polynomial in the size of the original network.*

Proof. The size of the extended network is $size(\overline{\mathcal{H}}) = size(\mathcal{H}) + 4 \cdot |V|$, as two half-edges are added to each vertex. For the expanded network, $\tilde{\mathcal{H}}$, the size depends on the in- and out-degree of the vertices in the extended network. Let $d_{\overline{E}}^-(v)$ denote the in-degree of $v \in V$, and $d_{\overline{E}}^+(v)$ the out-degree. Note that the degree counts the number of unique incident edges, so for $e \in \overline{E}, v \in V : m_v(e^-) > 1$ the size contribution of e to $d_{\overline{E}}^-(v)$ is still only 1. Then the size of the expanded network is

$$\begin{aligned} size(\tilde{\mathcal{H}}) &= size(\overline{\mathcal{H}}) - |V| + \sum_{v \in V} (d_{\overline{E}}^-(v) + d_{\overline{E}}^+(v)) + 3 \sum_{v \in V} d_{\overline{E}}^-(v) \cdot d_{\overline{E}}^+(v) \\ &\leq size(\overline{\mathcal{H}}) - |V| + 2 \cdot |V| \cdot |E| + 3 \cdot |V| \cdot |E|^2 \end{aligned}$$

where the inequality stems from the fact that at most all vertices are in all head and tail sets, in the original network. \square

Translation of Flow

Proposition. *A feasible flow $f: \tilde{E} \rightarrow \mathbb{N}_0$ on $\tilde{\mathcal{H}}$ can be converted into an equivalent feasible flow $g: \overline{E} \rightarrow \mathbb{N}_0$ in $\overline{\mathcal{H}}$, with: $g(e) = f(\tilde{e})$, for all $e \in \overline{E}$.*

Proof. If f is feasible, (9) holds for all $\tilde{v} \in \tilde{V}$. By the definition of $\tilde{\mathcal{H}}$, we can say that (9) holds for all $\tilde{v} \in V_v^- \cup V_v^+$ for all $v \in V$. Recall that all transit edges have singleton heads and tails, and $f(e) = f(\tilde{e}), \forall e \in \overline{E}$. Thus, by addition of (9) in each $v \in V$ we get

$$\begin{aligned} \forall v \in V \quad : \quad & \sum_{u_{v,e}^-, e \in V_v^-} \left(\overbrace{\sum_{u^+ \in V_v^+} f((u_{v,e}^-, u^+))}^{\text{out-flow of } u_{v,e}^-} - \overbrace{m_{u_{v,e}^-}(e^-) f(e)}^{\text{in-flow of } u_{v,e}^-} \right) \\ & + \sum_{u_{v,e}^+, e \in V_v^+} \left(\overbrace{m_{u_{v,e}^+}(e^+) f(e)}^{\text{out-flow of } u_{v,e}^+} - \overbrace{\sum_{u^- \in V_v^-} f((u^-, u_{v,e}^+))}^{\text{in-flow of } u_{v,e}^+} \right) = 0 \end{aligned}$$

Here, the flow along each transit edge is first added and then subtracted again, so we can simplify the expression to

$$\forall v \in V \quad : \quad \sum_{u_{v,e}^-, e \in V_v^-} -m_{u_{v,e}^-}(e^-) f(e) + \sum_{u_{v,e}^+, e \in V_v^+} m_{u_{v,e}^+}(e^+) f(e) = 0$$

Using the definition of V_v^- and V_v^+ (Eq. (7) and (8)) one verifies that these relaxed constraints are exactly those of Eq. (3), i.e., the constraints on flows in $\overline{\mathcal{H}}$. \square

Proposition. *Let $f: \overline{E} \rightarrow \mathbb{N}_0$ be a feasible flow on $\overline{\mathcal{H}}$. It can then be decided in polynomial time, in the size of \mathcal{H} , if a feasible flow $g: \tilde{E} \rightarrow \mathbb{N}_0$ in $\tilde{\mathcal{H}}$ exists such that $g(\tilde{e}) = f(e)$ for all $e \in \overline{E}$. If it exists it can be computed in polynomial time.*

Proof. The proof proceeds by a reduction to finding a feasible flow in bipartite normal directed graphs, with balance constraints. We refer to [26, 51] for a definition of this problem. Recall that the edges of $\overline{\mathcal{H}}$ are translated directly into a subset of the edges in $\tilde{\mathcal{H}}$, and we as such are tasked with finding a feasible flow on all the transit edges, which can be decomposed into finding a feasible flow for each expanded vertex independently. Let $v \in V$, then the hypergraph $(V_v^- \cup V_v^+, E_v)$ only contains edges with singleton head

and tail multisets. It is therefore a normal directed, bipartite graph. We then define the flow balance function $b: V_v^- \cup V_v^+ \rightarrow \mathbb{N}_0$ as $\forall u_{v,e}^- \in V_v^- : b(u_{v,e}^-) = f(e)$ and $\forall u_{v,e}^+ \in V_v^+ : b(u_{v,e}^+) = -f(e)$. Using the natural lower bound of flow $l \equiv 0$ and infinite upper bound finally gives us the complete specification. A feasible integer flow, if one exists, can be found in polynomial time in the size of the network [26, 51]. \square

The problem of finding a pathway with maximum production of specific compounds is known to be NP-hard, even for networks with bounded degree reactions [29]. The last proposition underlines that the expansion of the network does not drastically increase the complexity of finding pathways, but it also shows a potentially practical algorithmic approach to working with flows in the expanded network. In Sec. 4 we will however show a simpler approach to directly find the flows in the expanded network, using integer linear programming.

2.6 Chemical Transformation Motifs

The formal model of pathways as integer-valued hyperflows outlined above sets the stage for a concise analysis of large chemical reaction networks. From a chemical point of view the notion of a *chemical transformation motif* (CTM) takes on a central role in this context. It refers to a coherent subset of chemical reactions that have a well-defined interface to the remainder of the network and implements a discernible overall chemical transformation. Metabolic pathways and subnetworks networks exhibiting overall catalysis or overall autocatalysis are concrete instantiations of CTMs within a given chemical reaction network. It is useful, therefore, to think of a CTM as a formal specification of a set of integer hyperflows in a chemical reaction network, possibly in conjunction with additional constraints on the allowed hyperflows. A key property of CTMs is that different instantiations can replace each other in a given reaction network. CTMs therefore do not completely specify the chemical reactions in general. In the section on glycolysis pathways, for example, we will investigate in some detail the transformation motif *defined* by the conversion of 1 glucose to 3 acetylphosphates. The many ways of instantiating this motif will be our key concern.

In recent years the interest in CTMs resurfaced in the context of designing alternative networks performing the same function as their natural archetypes [60], as well as in the context of the notion of optimality in biochemical network structure. While the earliest work focused on small, well-characterised pathways such as the pentose phosphate pathway and the TCA cycle [42, 61], recent work extends the original approaches to larger networks including diverse reaction chemistry such as the central carbon metabolism [62] or the CO₂ fixation pathways [63]. In this literature the concept of CTM is only discussed implicitly and to our knowledge there has been no explicit attempt to formalize CTMs.

3 Comparison to Existing Methods

The basic pathway model described in Sec. 2.2 is quite similar to the formalism used in FBA, EFM and ExPa, with the latter two methods primarily aiming to categorise specific classes of pathways [64]. The basic model is also similar to the model presented in [24], although the specification of allowed I/O flow is phrased differently. In the following we will recast FBA in terms of hypergraphs as the underlying models of reaction networks to clarify the similarities but also the differences with our present approach.

The mathematical development of FBA, EFM, and ExPa is based upon the concepts of the *stoichiometric matrix* and *flux vectors*. These are analogous to a directed

multi-hypergraph and non-integer hyperflows. Let (\mathcal{H}, S, T) denote an I/O-constrained reaction network as defined in Sec. 2.2, with $\mathcal{H} = (V, E)$. The network can be represented as two matrices, an out-incidence matrix \mathbf{S}^+ and an in-incidence matrix \mathbf{S}^- , both in the domain $\mathbb{N}_0^{|V| \times |E|}$. That is, each row represents molecules and each column represents reactions. Let vertices and hyperedges have some arbitrary total order, $V = \{v_1, \dots, v_{|V|}\}$ and $E = \{e_1, \dots, e_{|E|}\}$. Then for each pair of vertices and reactions, v_i, e_j , the matrices are defined as $\mathbf{S}_{i,j}^+ = m_{v_i}(e_j^+)$ and $\mathbf{S}_{i,j}^- = m_{v_i}(e_j^-)$. Thus the columns of \mathbf{S}^+ represents the tail-multiset of each hyperedge, likewise for \mathbf{S}^- and the head-multisets. The actual stoichiometric matrix is defined as $\mathbf{S} = \mathbf{S}^- - \mathbf{S}^+$, which in chemical terms is the change of the number of each molecule that each reaction induces. The stoichiometric matrix describes both the proper chemical reactions and transport of material from and to the outside, equivalent to the extension of a hypergraph \mathcal{H} to an extended hypergraph $\overline{\mathcal{H}}$, see Eq.(2).

The stoichiometric matrix \mathbf{S} completely describes the original reaction network, and thus is equivalent to the I/O-constrained reaction network (\mathcal{H}, S, T) if and only if all hyperedges have disjoint head and tail. All direct catalysts, however, are cancelled out in the stoichiometric matrix, hence the equivalence fails whenever there are reaction hyperedges with $e^+ \cap e^- \neq \emptyset$. This somewhat limits the scope of FBA. Although it is possible in principle to replace reactions with direct catalysts by a sequences of intermediate reactions that consume and regenerate the catalyst, the resulting FBA network is no longer equivalent to the original one and allows the drainage of intermediates. This alters flux solutions and may be undesirable, e.g., when modelling the concerted action of enzyme complexes. Inspired by natural biosynthetic pathways industrial biocatalysis research [65,66] currently intensively investigates multi-enzyme cascade reactions, i.e. the combination of several enzyme reactions in concurrent one-pot processes [67], because of their prospect towards a “greener” and more sustainable chemical future. Intermediates from these cascades cannot be accessed as substrates by reactions outside the cascade; hence they require special treatment when represented explicitly.

A *flux vector* $f \in \mathbb{R}^{|E|}$ models a pathway, and must satisfy the usual conservation constraint, $\mathbf{S} \cdot f = 0$ (cmp. (3)). Reversible reactions are modelled in one of two ways:

- Combined: reversible reactions are modelled as a single reaction, but with the flow/flux allowed to be negative. The flow/flux of irreversible reactions is constrained to be non-negative. This is the approach followed when finding EFMs [68].
- Separate: reversible reactions are modelled as two inverse reactions, and the flow/flux on all reactions must be non-negative. This is the approach followed when finding ExPas [69]. We also follow this approach in our contribution both for mathematical simplicity and because it allows us to make use of the enhanced modelling capabilities offered by the expanded network.

The extension of the stoichiometric matrix \mathbf{S} to incorporate I/O reactions can also be implemented using both the “combined” and the “separate” way of handling reversible reactions. The I/O constraints from Eq. (4), specified by S and T , translate naturally to the corresponding constraints on the extended flux vector.

Linear Programming *versus* Integer Linear Programming With FBA we additionally define a linear objective function to find an optimal flux vector, possibly with additional linear constraints [70]. As a flux vector is real-valued, and all the stated constraints are linear, it can be found using linear programming (LP) in polynomial time [71,72]. Herein lies a major difference to the model presented in this contribution,

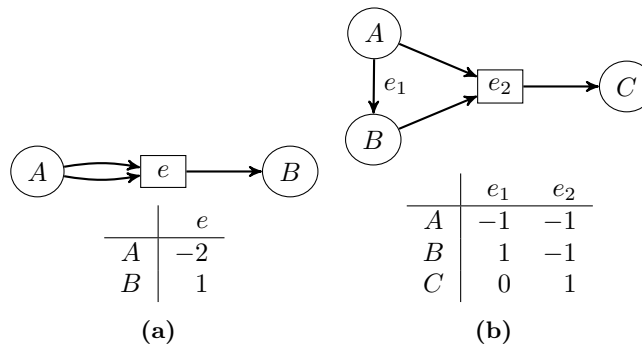


Figure 12. Examples of reaction networks with not totally unimodular stoichiometric matrices. (a) all entries in a TU matrix must be -1 , 0 , or $+1$. (b) the submatrix consisting of the top two rows has determinant 2.

Flow	Reaction
1.0	$\text{G3P} + \text{DHAP} \longrightarrow \text{FBP}$
1.0	$\text{G3P} \longrightarrow \text{DHAP}$
0.5	$\text{R5P} \longrightarrow \text{X5P}$
0.5	$\text{E4P} + \text{F6P} \longrightarrow \text{G3P} + \text{S7P}$
1.5	$\text{P}_i + \text{X5P} \longrightarrow \text{AcP} + \text{G3P} + \text{H}_2\text{O}$
0.5	$\text{P}_i + \text{F6P} \longrightarrow \text{AcP} + \text{E4P} + \text{H}_2\text{O}$
0.5	$\text{P}_i + \text{S7P} \longrightarrow \text{AcP} + \text{R5P} + \text{H}_2\text{O}$
1.0	$\text{FBP} + \text{H}_2\text{O} \longrightarrow \text{P}_i + \text{F6P}$
Overall	$\text{X5P} + 1.5 \text{P}_i \longrightarrow 2.5 \text{AcP} + 1.5 \text{H}_2\text{O}$

Table 1. A non-integer pathway with maximum production of AcP from 1 X5P. The optimal integer solution consists of the shaded reaction with flow 1. Relaxing the integrality constraint thus allows for a recycling pathway using G3P to produce 1.5 extra copies of AcP. See Sec. 5 for a table of molecule abbreviations.

where we require an *integer* hyperflow. We can thus characterise the linear program from FBA as the *LP relaxation* of the basic pathway problem presented here.

The LP relaxation of an ILP yield an integer solution only under special conditions. The best known sufficient condition is that the matrix of constraint coefficients is totally unimodular (TU), i.e., when all its square submatrices have determinants -1 , 0 , or $+1$, and thus all entries of the matrix are also -1 , 0 , or $+1$. This is the case for example for integer flows in graphs [26, 51]. As the simple examples in Fig. 12 shows, this not true in general for stoichiometric matrices and hence for hyperflows.

Even though total unimodularity is not a necessary condition, it is not too difficult to construct reaction networks with linear optimisation problems where the integer problem and the LP relaxation have drastically different optimal solutions. As an example consider the carbon rearrangement network described later, in Sec. 5.1, and the question: given 1 xylulose 5-phosphate (X5P) and an arbitrary amount of phosphate (P_i), find a pathway that maximises the production of acetylphosphate (AcP). As X5P contains 5 carbon atoms and AcP contains 2, it is clear that the maximum production from a single molecule must be at most 2 AcP. It turns out that the optimal integer solution just 1 AcP, by the single reaction $\text{P}_i + \text{X5P} \longrightarrow \text{AcP} + \text{G3P} + \text{H}_2\text{O}$. However, the optimal solution to the LP relaxation of the problem yields 2.5 AcP, via the pathway described in Tab. 1. This non-integer pathway incidentally uses the same reaction as the integer pathway, with an additional recycling network for converting 1 G3P into 1.5 AcP.

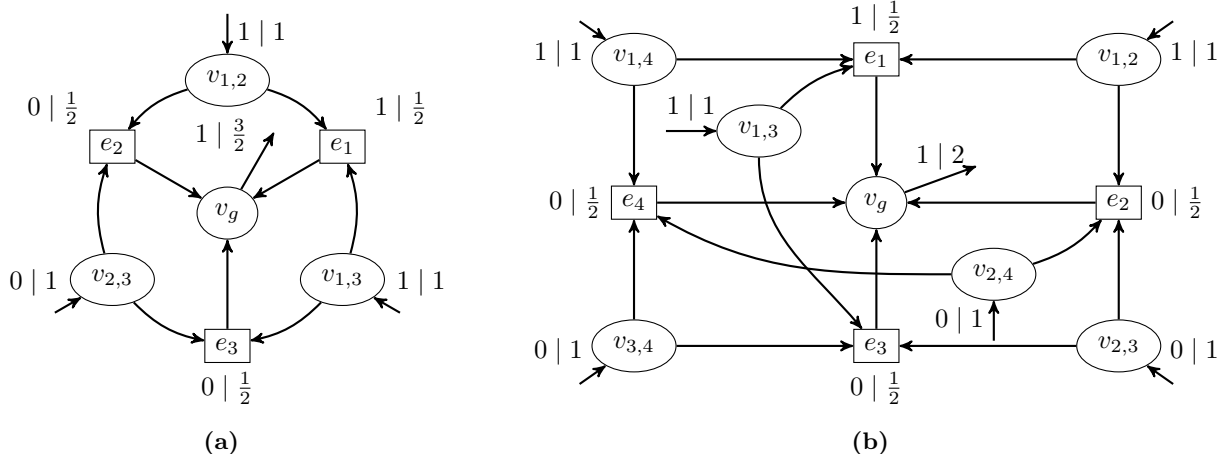


Figure 13. Reduction from the independent set problem to the problem of maximising output from a molecule in a reaction network, applied to the two graphs (a) K_3 and (b) K_4 . The hyperedges are annotated with both a feasible integer flow and a feasible non-integer flow, written as $\langle \text{integer} \mid \text{non-integer} \rangle$. Allowing a maximum inflow of 1 to all vertices $v_{i,j}$ and maximising the outflow of v_g corresponds to finding a maximum independent set in the original graph.

A scaling of the non-integer flow with a factor 2 gives an integer solution, and indeed our methods can also find this as well as many other solutions implementing the same motif. Since LP solutions with integer-valued constraint matrices and objective functions with integer coefficients are rational, it is mathematically always possible to scale the LP solution to integer values. The actual numbers, however, may become very large. Taking physiological constraints into account, the number of available individual molecules may be small, even as low as 100 copies [73], and even smaller for biological macromolecules [74, 75]. Small numbers may have a profound influence on chemical kinetics and in many cases can cause qualitative changes in the system’s behaviour [76–78]. It is an important feature of the ILP framework to allow for directly expressing such biological constraints. At the same time, it provides pathway solutions that have a direct mechanistic interpretation.

Integrality Gap In the previous example we maximised the production of a specific molecule, and saw that the ILP solution have objective value 1 and the LP relaxation have objective value 2.5. The ratio between these values is known as the *integrality gap*, and it is known that this gap can scale with the input instance. For a simple example, consider the reaction networks stemming from the polynomial-time reduction described in the supplemental material S1, reducing the well-known INDEPENDENT SET problem to maximising the production of a single molecule in a reaction network. Applying the reduction to complete graphs with n vertices and formulating the problem in terms of hyperflows, we obtain an integrality gap of $\frac{n}{2}$: for integer flows we can use at most 1 reaction, thus giving a maximum output of 1. When the integrality constraint is removed we can let the flow be 0.5 on all reactions, giving an output of $\frac{n}{2}$. The reaction networks for the complete graphs of size 3 and 4 are shown in Fig. 13. This illustrates that the use of the LP relaxation is not just a technical detail, but changes the nature of the problem entirely. We remark that the problem of finding an integer hyperflow with maximum production of a given compound has been proven strongly NP-hard [29], even for the restriction to reaction networks with 2-to-2 reactions.

Solution Enumeration A linear program may have an uncountable number of optimal solutions as the variables are in the domain of real numbers. The solutions can, however, be described by enumerating the, possibly exponentially many, corners of the optimal face of the polyhedron defined by the linear program [79, 80]. This has also been applied to FBA [81].

In Sec. 4.4 below we describe a simple method for enumerating solutions when using integer linear programming. When restricting the solutions to be integer, we only have finitely many candidates, and it is straight-forward to enumerate not only optimal solutions, but also near-optimal solutions. This is of relevance in particular in design applications. Here alternative co-optimal or near optimal solutions may be easier to realize in practise than a particular optimal pathway.

4 Implementation using Integer Linear Programming

The ILP formulation characterising feasible flows is based on an expanded hypergraph $\tilde{\mathcal{H}} = (\tilde{V}, \tilde{E})$. The flow function is modelled by an integer variable x_e for each edge $e \in \tilde{E}$, and by constraints for flow conservation. The basic constraints are thus

$$\begin{aligned} \sum_{e \in \delta_{\tilde{E}}^+(v)} m_v(e^+) \cdot x_e - \sum_{e \in \delta_{\tilde{E}}^-(v)} m_v(e^-) \cdot x_e &= 0 & \forall v \in \tilde{V} \\ x_e &\in \mathbb{N}_0 & \forall e \in \tilde{E} \end{aligned}$$

This definition is similar to an ILP formulation of a classical network flow problem, but with important differences; $\tilde{\mathcal{H}}$ is a hypergraph so an edge $e \in \tilde{E}$ may be in both $\delta_{\tilde{E}}^-(u)$ and $\delta_{\tilde{E}}^-(v)$ (or $\delta_{\tilde{E}}^+(u)$ and $\delta_{\tilde{E}}^+(v)$) for $u \neq v$. Additionally, $\tilde{\mathcal{H}}$ is a *multi*-hypergraph, and thus the coefficients $m_v(e^+)$ and $m_v(e^-)$ are introduced, which may be larger than 1.

The basic formulation can be augmented with constraints, e.g., on the I/O edges, and an objective function depending on the specific problem to be solved. Additionally, the constraints for chemical flows specified in Eq. (10) are added in the obvious way. In the following sections we describe constraints for finding catalytic and autocatalytic flows. For the formulation we will use M to denote a classical “large enough” constant, and as some parts of the catalysis and autocatalysis models are similar we will first describe the formulation of these common parts.

4.1 Strict Flow Through Overall (Auto)catalytic Vertices

In our definition of (auto)catalysis we require that if a vertex is (auto)catalytic, then no flow can enter the vertex from the network and exit the vertex to the network again. Let z_v be the indicator variable for the vertex $v \in V$ being (auto)catalytic, then the requirement is trivially enforced by the following constraints:

$$x_e \leq M \cdot (1 - z_v) \quad \forall e = (u_{v,e'}^-, u_{v,e''}^+) \in V_v^- \times V_v^+ : e' \neq e^- \wedge e'' \neq e^+$$

4.2 Overall Catalysis

We model catalysis by introducing an indicator variable $z_v^c \in \{0, 1\}$ for each $v \in V$ indicating whether v is catalytic or not. Thus we can enforce a solution to be catalytic by posing the constraint

$$\sum_{v \in V} z_v^c \geq 1$$

The actual constraints for the indicator variables are obtained partially by the section above on strictness of flow. Below follows the last requirement, Eq. 5, which is realised through a set of auxiliary indicator variables, $z_v^0, z_v^<, z_v^> \in \{0, 1\}$

$$\begin{aligned}
x_v^- = x_v^+ = 0 &\Leftrightarrow z_v^0 = 1 \equiv \begin{cases} 1 - z_v^0 \leq x_v^- + x_v^+ \\ M \cdot (1 - z_v^0) \geq x_v^- + x_v^+ \end{cases} \\
x_v^- < x_v^+ &\Leftrightarrow z_v^< = 1 \equiv \begin{cases} x_v^- < x_v^+ + M \cdot (1 - z_v^<) \\ x_v^- \geq x_v^+ - M \cdot z_v^< \end{cases} \\
x_v^+ < x_v^- &\Leftrightarrow z_v^> = 1 \equiv \begin{cases} x_v^+ < x_v^- + M \cdot (1 - z_v^>) \\ x_v^+ \geq x_v^- - M \cdot z_v^> \end{cases} \\
0 < x_v^- = x_v^+ &\Leftrightarrow z_v^c = 1 \equiv \begin{cases} z_v^c \geq 1 - z_v^< - z_v^> - z_v^0 \\ z_v^c \leq 1 - z_v^0 \\ z_v^c \leq 1 - z_v^< \\ z_v^c \leq 1 - z_v^> \end{cases}
\end{aligned}$$

4.3 Overall Autocatalysis

As for catalysis we model autocatalysis with a set of indicator variables $z_v^a \in \{0, 1\}$ for all $v \in V$, and force a solution to autocatalytic with the constraint

$$\sum_{v \in V} z_v^a \geq 1$$

We use the constraints for strictness of flow and model the remaining constraint, Eq. 6, using the auxiliary variable set z_v^- , indicating $x_v^- > 0$:

$$\begin{aligned}
0 < x_v^- &\Leftrightarrow z_v^- = 1 \equiv \begin{cases} z_v^- \leq x_v^- \\ M \cdot z_v^- \geq x_v^- \end{cases} \\
0 < x_v^- < x_v^+ &\Leftrightarrow z_v^a = 0 \equiv \begin{cases} z_v^a \leq x_v^- \\ x_v^- < x_v^+ + M \cdot (1 - z_v^a) \\ M \cdot z_v^a + x_v^- \geq x_v^+ - M \cdot (1 - z_v^-) \end{cases}
\end{aligned}$$

4.4 Solution Enumeration

A typical use of solvers for integer programs is to find a single optimal solution. However, from a chemical perspective we are also interested in near-optimal solutions and in some cases even all solutions. The structure of our formulation additionally have influence on when two solutions are considered different. Often we might not consider two solutions different if they only differ in the flow on the transit edges, i.e., those introduced by the vertex expansion. This makes it difficult to use build-in features in solvers, such as the solution pool in IBM ILOG CPLEX, to enumerate solutions.

For finding multiple solutions we therefore explore a search tree based on the domain of the variables; each vertex in the tree represents a restriction of the variable domains, with children representing more constrained domains. Note that this tree, in theory, is infinite as some variable may have no upper bound. In each vertex we use an ILP solver to find an optimal solution for the sub-problem. If the problem is infeasible the sub-tree is pruned, otherwise a path to a leaf in the tree is constructed to represent the solution found by the ILP solver. The quality of the found solution at the same time acts as a lower bound on the objective function of the sub-tree (when minimising the function). Vertices in the tree are explored in order of increasing lower bound.

If a different value of flow is not to be considered a difference in the solution we simply do not consider the corresponding variables to be part of the branching procedure.

4.5 Software Implementation

The pathway model is implemented as an extension of the larger software package, MedØlDatschgerl (MØD) [82,83]. The software combines the pathway analysis with methods for working with generative chemistries [22,84], including the algorithms for network expansion [23] also used in this contribution. The tight integration between the methods makes it a convenient tool to design artificial chemistries, both high-level systems like DNA-templated computing [85], or hypothetical prebiotic chemistries [86,87]. Our implementation uses IBM ILOG CPLEX 12.5 for solving integer linear programs. An upcoming version of MØD is in preparation that will include the pathway model.

5 Results

In the following we will illustrate the strength of our constructive approach with the help of the following problems. (1) find an “optimal” pathway transforming educt to product molecules using a predefined set of chemical transformations. What optimality means in this context will be discussed later. This type of problem is central to metabolic engineering in the Synthetic Biology context, where frequently design objectives and reactions in the form of enzymes are given but no network to optimise on! For the solution clearly constructive chemistry approaches which generates the network during the optimisation are indispensable. (2) find sub-networks, embedded in a large chemical reaction network, which optimise a given objective. Besides well established objectives such as the minimization of the number of reactions used or maximizing the yield of a target compound, the objective can also be a chemical transformation pattern for example autocatalysis $F + X \longrightarrow 2 X + W$. Here F and W are arbitrary “food” and “waste” molecules needed during the self-reproduction process of the autocatalyst X. Of course for the later case the identified subnetwork possess only the right topological requirements to exhibit autocatalytic behaviour. No statements about the kinetic properties are made.

Evolution resulted in the emergence of complex life forms, showing a remarkable adaptation to the environmental conditions of their habitats. Natural selection was identified as the major driving force behind this intricate optimisation process, that shapes the structure and logic of functional entities on all scales of biological organization. A specific system of interest is cellular metabolism the chemical machinery which supports every biological function. Metabolism is usually understood as a collection of enzyme-catalysed reaction sequences tied into a network by common intermediate metabolites. By encapsulating the reaction chemistry into enzymes, i.e., modules operating under almost identical conditions, nature achieved a lego-type plug-and-play system for the physical implementation of nearly any desired overall reaction mechanism via chaining of the appropriate enzymes. It is believed, that the design principles and shaping constraints of modern cellular metabolism can be elucidated via optimality analysis [61]. In particular since the same overall conversion of matter can be achieved by different not necessarily overlapping sets of enzymes, it seems natural to view a given metabolic pathway as solution of a combinatorial optimisation problem.

The three chemistries are all modelled using the graph grammar approach described in the introduction, and the reaction networks are discovered using exploration

strategies. A visualisation of all rules can be found in the supplemental material S3, along with tables of molecule name abbreviations S2.

5.1 Carbon Rearrangement in the Non-oxidative Glycolysis Pathway

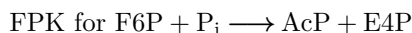
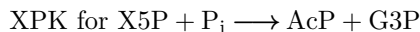
Finding sustainable processes for the produce of fuel molecules, that drive our modern society, is one of the big challenges today. One possibility, which is intensively explored, is to engineer the biochemical setup of microbes, to convert feedstock molecules, such as glucose, into the desired fuel molecules. The biochemical idea behind this approach is to couple the catabolic pathway known as glycolysis, which decomposes glucose (a C₆ molecule) into acetyl coenzyme A units (AcCoA, a C₂ molecule), to a designed synthetic pathway, condensing AcCoA into the skeleton of the target molecule. The drawback of this approach is, however, that 2 of the 6 carbon atoms from glucose are lost as CO₂ during normal oxidative glycolysis, which pushes the yield of this pathway down to 75%, in terms of atom economy. Coupling the biofuel producing synthetic pathway to a lossy process for educt production is, from the economic perspective, a bad idea.

A recent study [44] attacks the aforementioned problem by hand crafting a non-oxidative glycolysis (NOG) pathway, which prevents the carbon atom loss of its natural counterpart. The general logic of this designed pathway is to couple the splitting reaction that produces the desired C₂ body and a C₄ body as putative waste, to a carbon rearrangement network, which then recycles the C₄ body into molecules, that can be fed back into the NOG as educts. With this strategy NOG achieves a 100% carbon atom economy. The architecture of metabolic networks in general shows this kind of self-referential topology, every putative waste molecule is recycled, making metabolic networks very different from unevolved chemical reaction networks (e.g., atmosphere or geochemical networks), which do not show this property.

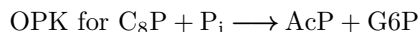
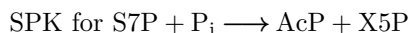
The paper [44] discusses several sources of variation for the structure of the NOG pathway. First the splitting reaction can be performed by two types of phosphoketolase (PK) enzymes, differing only in their input sugar preference; either fructose (F) or xylulose (X). Second the carbon rearrangement network can go either via fructose 1,6-bisphosphate (FBP, a C₅ sugar) or sedoheptulose 1,7-bisphosphate (SBP, a C₇ sugar). This freedom allows for three basic structural designs (see Fig. 2, [44]) of the NOG pathway. In this section we illustrate that many more equivalent solutions can be found automatically. We use a generic model of the chemistry in order to explore related reactions for which concrete enzymes may not yet exist.

Modelling The molecules are encoded as graphs in the straight-forward manner, though without stereochemical information. This implies that certain classes of molecules are represented as a single molecule, e.g., Ru5P and X5P.

We have modelled the generic transformation rules listed in Tab. 2, and shown in the supplemental material S3. In [44] the use of phosphoketolase is associated with specific names for the specific reactions:



We extend the naming scheme to cover educts with 7 and 8 carbons:



To create the reaction network we use the starting molecules P_i, AcP, G3P, DHAP, E4P, R5P, Ru5P, F6P, S7P, and FBP.

Abbr.	Name	Description	Example Reaction
AL	Aldolase	A generic aldol addition.	$G3P + DHAP \longrightarrow FBP$
AlKe	Aldose-Ketose	Aldehyde to ketone conversion.	$R5P \longrightarrow Ru5P$
KeAl	Ketose-Aldose	Ketone to aldehyde conversion.	$Ru5P \longrightarrow R5P$
PHL	Phosphohydrolase	Use water to cleave off phosphate.	$FBP + H_2O \longrightarrow F6P + P_i$
PK	Phosphoketolase	Break C-C-bond and add phosphate.	$F6P + P_i \longrightarrow AcP + E4P$
TAL	Transaldolase	Move C ₃ -end.	$F6P + E4P \longrightarrow G3P + S7P$
TKL	Transketolase	Move C ₂ -end.	$G3P + S7P \longrightarrow X5P + R5P$

Table 2. List of generic transformation rules for modelling the non-oxidative glycolysis chemistry. The full details of the rules are shown in the supplemental material S3.

Network The network is created by iterating the application of all the transformation rules listed in Tab. 2, until no new molecules are discovered. This chemistry is theoretically infinite, so we impose the restriction that no molecule with more than 8 carbon atoms may be created. This is a reasonable constraint, since even under physiological conditions carbohydrates are inherently metastable compounds. For carbohydrates larger than C₈ the decomposition reactions become a dominating reaction channel. Although alternatives (L-type PPP) or extended reaction sequences via higher carbohydrates have been suggested their biochemical evidence has been questioned (see discussion in recent review [88]). The network generation therefore terminates, and results in a network with 81 molecules and 414 reactions.

Pathways For the overall reaction $F6P + 2P_i \longrightarrow 3AcP + 2H_2O$ we have enumerated all solutions using at most 8 unique reactions and with at most 11 reactions happening in total. Additionally we disable the AL and PK reactions with small educt molecules; those with less than 3 carbon atoms each. This is in agreement with the experimentally characterised reversible ping-pong mechanism of these two enzymes [43]. This results in 263 different pathways, which were computed within a few minutes on a normal desktop computer. In Tab. 3 we categorise the pathways according to the properties

- number of unique reactions used,
- number of reactions used,
- whether the only bisphosphate used is FBP,
- the histogram of different PK reactions (see the modelling above).

The table shows the number of solutions in each combination. Interestingly it turns out that the solution space where FBP is the only bisphosphate used is quite similar to the space where other bisphosphates are allowed but with only 7 unique reactions. The solutions are distributed in the same manner except for a 1-shift in the number of reactions. There is a 1-to-1 mapping between these two sets of solutions such that the only difference in the pathway is the sub-pathways described in Tab. 4.

In Fig. 14 one of the solutions are illustrated in detail. This solution has similar properties to the solution shown in Fig. 2a in [44]: the phosphoketolase reactions all have F6P as educt, and the only bisphosphate used is FBP. However, this solution can be regarded as being shorter as it uses fewer reactions, though the number of unique reactions is the same. Allowing other bisphosphates than FBP to be used enables even shorter solutions to be found. Fig. 15 shows the shortest solution, which uses a bisphosphate with 7 carbon atoms. Its use of phosphoketolase is also different, as it uses both XPK, FPK, and SPK.

Phosphoketolase Type XPK, FPK, SPK, OPK	Only FBP				Other Bisphosphates								
	8 Unique Reactions				7 Unique Reactions					8 Unique Reactions			
	Reactions				Reactions					Reactions			
	8	9	10	11	7	8	9	10	11	8	9	10	11
0, 0, 0, 3	-	-	-	-	-	-	-	-	-	-	-	4	16
0, 0, 1, 2	-	-	-	-	-	-	-	-	-	-	3	2	-
0, 0, 2, 1	-	-	-	-	-	-	-	-	-	-	4	-	-
0, 0, 3, 0	-	-	1	2	-	-	1	2	-	-	-	9	20
0, 1, 0, 2	-	-	-	-	-	-	-	-	-	-	4	4	-
0, 1, 1, 1	-	-	-	-	-	-	-	-	-	3	-	-	-
0, 1, 2, 0	-	1	-	-	-	1	-	-	-	-	8	2	-
0, 2, 0, 1	-	-	-	-	-	-	-	-	-	-	6	-	-
0, 2, 1, 0	-	1	-	-	-	1	-	-	-	-	9	-	-
0, 3, 0, 0	-	-	2	4 _a	-	-	2	4	-	-	-	14	24
1, 0, 0, 2	-	-	-	-	-	-	-	-	-	-	2	4	-
1, 0, 1, 1	-	-	-	-	-	-	-	-	-	1	-	-	-
1, 0, 2, 0	-	1	-	-	-	1	-	-	-	-	6	2	-
1, 1, 0, 1	-	-	-	-	-	-	-	-	-	2	-	-	-
1, 1, 1, 0	1	-	-	-	1	-	-	-	-	3	-	-	-
1, 2, 0, 0	-	2	-	-	-	2	-	-	-	-	10	-	-
2, 0, 0, 1	-	-	-	-	-	-	-	-	-	-	4	-	-
2, 0, 1, 0	-	1	-	-	-	1	-	-	-	-	7	-	-
2, 1, 0, 0	-	2 _c	-	-	-	2	-	-	-	-	10	-	-
3, 0, 0, 0	-	-	2 _b	4	-	-	2	4	-	-	-	12	20

Table 3. Overview of number of NOG pathways. Categories marked with subscript a , b , and c refer to Fig. 2 of [44], and we see that not only are there alternate solutions in the exact same categories, but in the case of a we even find a shorter pathway with the same properties. Note that the left block is similar to the middle block of categories, but with 1 less reactions used. This is due to a replacement of part of the pathway with a shorter pathway, see Tab. 4. The pathway shown in Fig. 14 is from the framed blue category, and the pathway in Fig. 15 is from the framed green category. Their counterparts with the replacement from Tab. 4 are the unframed shaded numbers.

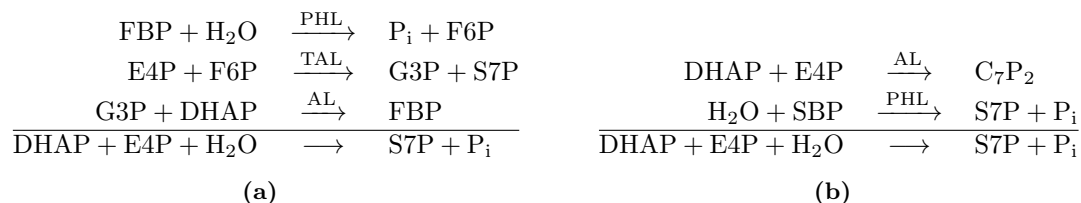


Table 4. Two pathways with the same overall reaction. (a) a 3-reaction pathway using FBP as bisphosphate. This sub-pathway is highlighted in Fig. 14. (b) a 2-reaction pathway using a bisphosphate with 7 carbons. This sub-pathway is highlighted in Fig. 15.

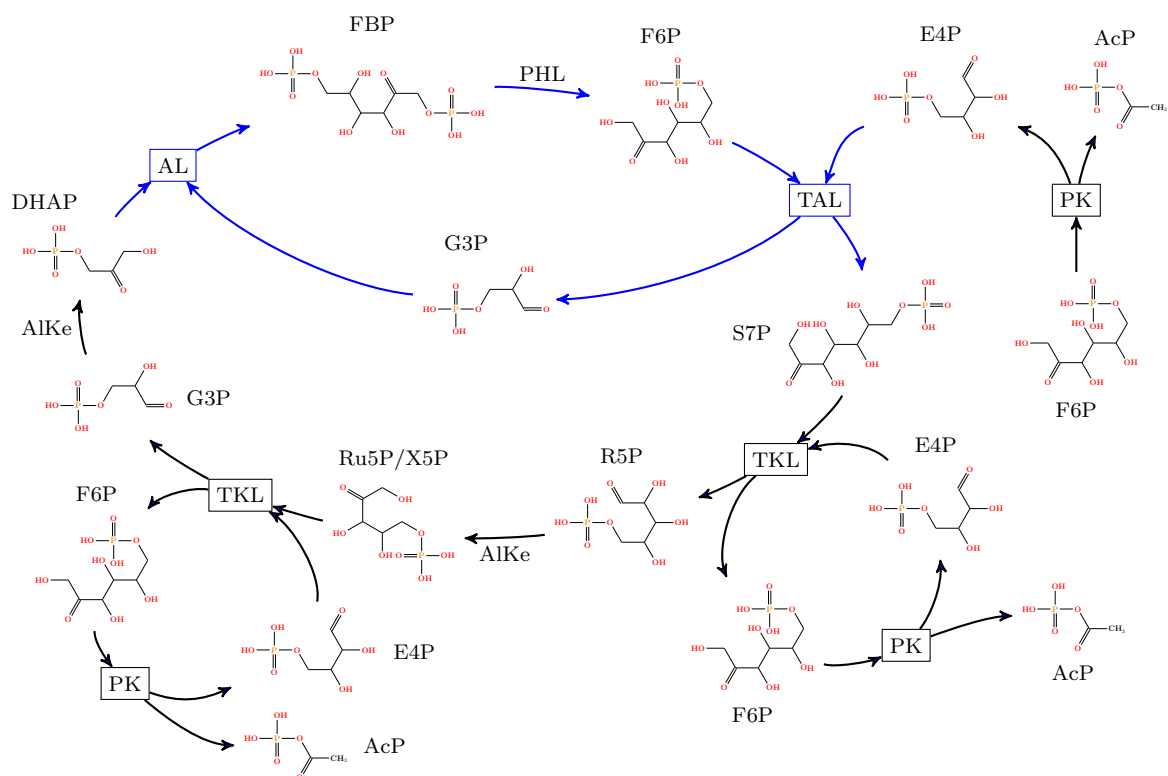


Figure 14. Illustration of a pathway similar to the one depicted in Fig. 2a of [44], using fewer reactions. This solution category is the framed blue cell of Tab. 3. The highlighted sub-pathway is the pathway from Tab. 4a.

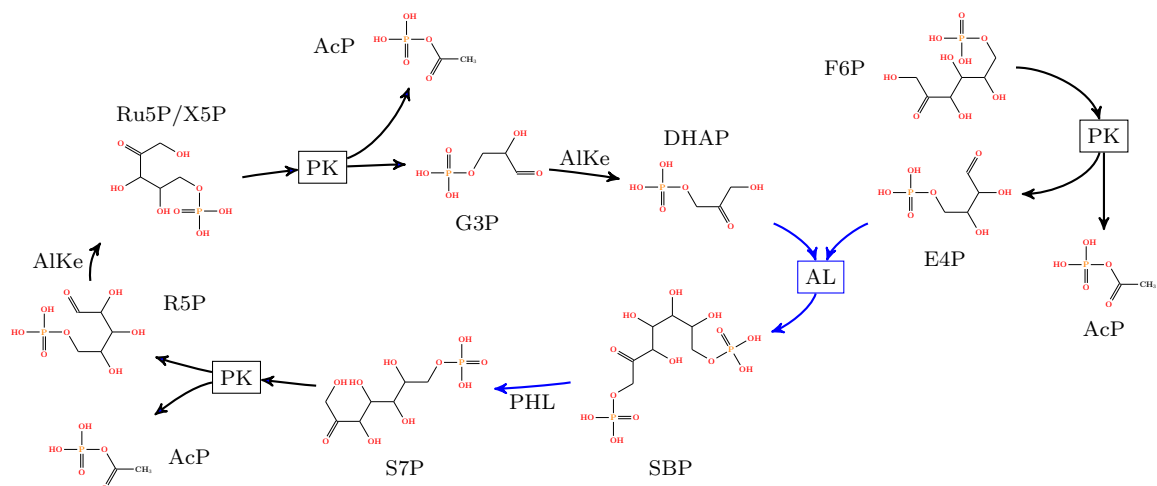


Figure 15. Illustration of the shortest NOG pathway, denoted by the framed green cell in Tab. 3. It uses three different phosphoketolase reactions: XPK, FPK, and SPK. The highlighted sub-pathway is the pathway from Tab. 4b.

The basic structure of the NOG solutions combine a productive splitting reaction with a recycling network, which converts the “waste” produced during the splitting reaction back into starting material using carbon rearrangements. The major source of variability in the solutions stem from the flexibility and combinatorial nature of the carbon rearrangement chemistry. Our investigation nicely illustrates how vast the chemical network design space is. Even if the reaction chemistry is restricted to only a handful of enzyme functionalities, systematic exploration of the network design space without computational approaches is inefficient and many interesting solutions may be missed.

5.2 The Citric Acid Cycle and Glyoxylate Shunt

The citric acid cycle (TCA) is found with slight variations [89] in all kingdoms of life. One of TCA’s basic functions is to provide redox-equivalents and H^+ to the respiratory chain to harvest energy in the form of ATP molecules. Furthermore, the TCA is one of the hubs in central carbon metabolism which interfaces between catabolic and anabolic pathways which provide the necessary building blocks for cellular functions. Two carbon compounds in the form of acetyl-CoA are fed into the TCA, where they are merged with a four carbon compound to yield citric acid (a type of tricarboxylic acid), which gave the cycle its name. Subsequently two carbon atoms are split off from the citric acid as CO_2 followed by a series of “recycling” reactions to regenerate the initial four-carbon compound, which finally closes the reaction sequence to a cycle. The acetyl-CoA, which is metabolised in the TCA, is either produced from breakdown of sugars via glycolysis or by degradation of fatty acids during β -oxidation. In bacteria, plants, and fungi a shortcut of the TCA exists, called the glyoxylate shunt, which allows these organisms to *de-novo* synthesize sugars from acetyl-CoA stemming from β -oxidation of fatty acids. This type of gluconeogenesis is possible since the glyoxylate shunt bypasses those steps in the TCA, where carbon is lost in the form of CO_2 . The glyoxylate shunt has been found in higher animals only up to the clade of nematodes [90]. Recently the question of gluconeogenesis from fatty acids via the TCA has been re-investigated using elementary mode analysis [91]. The results support the five decade old finding, that this conversion is only possible via the TCA if the glyoxylate shunt is included.

In the following we re-investigate the TCA cycle including the glyoxylate shunt and the feeding reactions to illustrate that the various operation modes (e.g., maximizing the production of redox-equivalents, autocatalytic solutions, etc.) of the TCA correspond to different objective function functions. In particular our approach identifies the gluconeogenesis from fatty acids as an autocatalytic solution in oxaloacetate, giving a mechanistic explanation why this TCA mode works, insight which cannot be gained directly by elementary mode analysis.

Modelling We use reaction patterns for the reactions in the TCA cycle and the glyoxylate cycle, and the patterns necessary to connect pyruvate with oxaloacetate and acetyl-CoA. We also include the inverse patterns, so we can generate the inverse cycles.

For the starting graphs we use a set of food molecules (e.g., water, NAD^+ and coenzyme-A). Additionally we add glyoxylate, pyruvate, acetyl-CoA, and all the central molecules of the TCA cycle.

Network The network is created from a 1-step expansion from the starting molecules. The central part of the network with the TCA and glyoxylate cycle is shown in Fig. 16.

Overall (Auto)catalytic Cycles For all queries we use oxaloacetate and the food molecules as the input set, and the set of all molecules as output set.

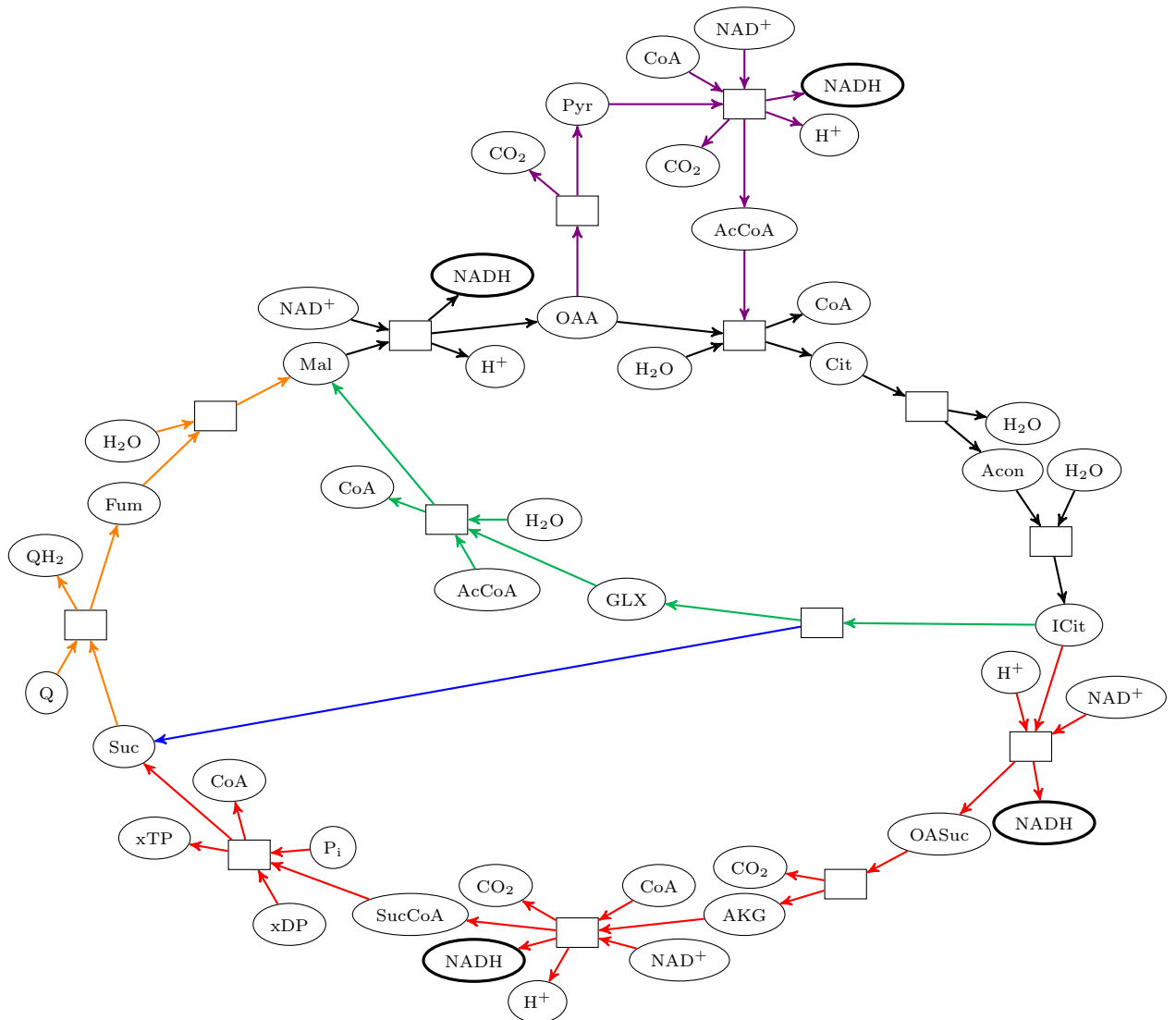


Figure 16. Abstract visualization of the central part of the TCA and glyoxylate network. The black, red, and orange parts correspond to the TCA cycle, which in the generated network is the catalytic cycle with the maximum production of NADH. Instead, minimising the length of the cycle leads to the glyoxylate cycle, shown in black and green. Searching of an autocatalytic cycle leads to two different possibilities: one based on the glyoxylate cycle (black, green, blue, orange), and one using the reverse TCA cycle (reverse black, red, orange, violet).

Using the objective of minimising the number of used reactions, we find a subset of the glyoxylate cycle. This subset is shown in black and green in Fig. 16. When we instead use an objective function which maximises the production of NADH, we find the TCA cycle, depicted in black and red in Fig. 16. Additionally, when searching for autocatalytic cycles we find the complete glyoxylate cycle, marked with black, green, blue, and orange in Fig. 16. Another found autocatalytic cycle is the reverse TCA cycle, obtained as the reverse of the black, red, orange, and violet parts of Fig. 16.

Results For demonstration purposes, we applied optimisation techniques, which are well established in the field of linear programming (to which FBA belongs as well). In particular, the reaction network was kept fixed while the objective function was varied to find solutions respecting specific side conditions. For example, finding autocatalytic solutions while minimizing the number of used reactions. If the network is queried for the shortest autocatalytic solution, a cyclic solution in oxaloacetate is found, which involves the glyoxylate shunt. This autocatalytic solution feeds on acetyl-CoA to produce two copies of oxaloacetate. One of them can be pushed up glycolysis, while the other copy maintains the autocatalytic production of more oxaloacetate from acetyl-CoA. This solution nicely explains mechanistically why organisms, which possess the glyoxylate shunt, can produce sugars from fatty acids. The reductive (reverse) TCA cycle is found as an alternative solution. It is one of six naturally occurring carbon fixation pathways (for recent reviews see [92, 93]), which is operated solely by bacterial lineages in anaerobic, high CO₂ environments. It requires only two ATP molecules to reduce CO₂ to pyruvate, and is therefore more ATP-efficient than the Calvin-Benson cycle [63]. In the context of origin of life research it was suggested that, the reductive TCA operated non-enzymatically on primitive Earth [94–96]. Its stability under prebiotic conditions is however highly debated [97].

5.3 Autocatalytic Cycles in the Formose Process

Carbohydrates (CH₂O)_n, vulgo sugars, can formally be viewed as polymers of formaldehyde CH₂O building blocks. The chemical reactivity of sugars is dominated by the two functional groups (1) carbonyl group (C = O) and (2) vicinal hydroxyl groups (HO – C – C – OH), making carbohydrates amenable to keto-enol tautomerization, aldol addition, and retro-aldol fragmentation reactions. Due to this intrinsic reactivity, sugars are potentially labile compounds, which readily isomerize into complex mixtures under non-neutral conditions. The formose process, described by Butlerov [98] is one of the scarce examples of an autocatalytic reaction network, which generates complex mixtures of sugars from an aqueous formaldehyde solution under high-pH conditions (for a recent review on autocatalysis see [53]). The formose process has intensively been studied (for a recent review see [99]) for its potential to produce biologically significant carbohydrates from formaldehyde under prebiotic conditions [100]. The time-concentration behaviour of formaldehyde consumption during the formose process shows a linear lag phase, followed by exponential consumption, and a levelling off when the formose processes runs out of formaldehyde supply. This is the point where the clear reaction mixture starts to turn yellow and the generated C₄–C₆ sugars isomerize to a combinatorially complex mixture of compounds and black tar. The core autocatalytic cycle of the formose process usually found in the literature [50, 99, 101], glycolaldehyde “fixates”, via a series of keto-enol tautomerizations and aldol additions, two formaldehyde molecules (thereby doubling in size) followed by a retro-aldol fragmentation, resulting in two copies of glycolaldehyde. However, experimental evidence exists [101, 102] that this base cycle cannot account for the massive consumption of formaldehyde, after the lag-phase. The reasons is that, under the

Abbr.	Name	Description	Example Reaction
KeEn	Keto-Enol	Keto to enol form conversion.	$C_{2a} \longrightarrow C_{2e}$
EnKe	Enol-Keto	Enol to keto form conversion.	$C_{2e} \longrightarrow C_{2a}$
AA	Aldol addition	Merge an enol and a keto.	$C_{2e} + C_1 \longrightarrow C_{3a}$
RAA	Retro-aldol add.	Split a keto into enol and keto.	$C_{3a} \longrightarrow C_{2e} + C_1$

Table 5. List of generic transformation rules for modelling the formose chemistry.

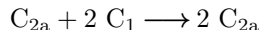
reaction conditions, enolization of the carbonyl group and aldol addition are much faster than “ketonization” (restoring the carbonyl group) required to close the autocatalytic base cycle. From this experimental evidence the following mechanistic picture arises; fast repeated addition of formaldehyde to enolized keto groups produces larger sugars, draining material from the the base cycle and retro-aldol fragmentation of larger sugars replenishes the base cycle (or variants) with short carbohydrates. How these higher order cycles are organized around the base autocatalytic cycle is unknown as well as their concrete structural organization and interconnectedness.

Modelling For the formose chemistry we will adopt the following naming scheme for molecules; $C_{N\langle t \rangle}$, where N specifies the number of carbon atoms and $\langle t \rangle$ indicates the position of the double bond. We use _a for aldehydes, _e for enol forms, and _k for ketones.

The chemistry is modelled based on [100] which describes two basic reactions: keto-enol tautomerism and aldol reaction. Both are reversible, thus giving four transformation rules listed in Tab. 5, and shown in the supplemental material S3. The starting molecules are formaldehyde (C_1) and glycolaldehyde (C_{2a}).

Network The network is expanded to include all derivable molecules with at most 9 carbon atoms, thus reaching a size of 284 molecules and 978 reactions. The computation time is a few seconds.

Enumeration of Solutions In the network we have enumerated overall autocatalytic flows starting from those of minimum size. Specifically, flows with the overall reaction



with minimum number of unique reactions used. For the purpose of this enumeration we consider two solutions equivalent if the set of reactions used is the same, thus how many times a reaction is used is ignored. Tab. 6 shows the resulting number of solutions found, grouped by the number of reactions used and the maximum size of molecules involved. The enumeration was split into 6 queries, one for each row of the table, and the combined computation time was approximately 134 hours. We were not able to find all the solutions corresponding to the two unknown entries in the table. Each of the queries, in the current implementation, needs more than 200 hours of computation time and more than 64 GB memory.

Results The computational analysis of the formose process chemical space reveals that the density of autocatalytic cycles is very high. The majority of the enumerated autocatalytic cycles involve higher sugars (C_5 - C_8), but conform to the overall reaction of the shortest possible autocatalytic cycle, referred to as base cycle. The higher cycles branch off from compounds in the base cycle and merge back to the base cycle further downstream. The resulting structure of interwoven autocatalytic cycles is highly self-referential and shows, with respect to this property, similarities to evolved metabolic networks, where also all tentative waste compounds are recycled by feeding them back

Unique reactions used	Maximum #C						Sum
	4	5	6	7	8	9	
6	0	0	1	1	1	2	5
7	0	0	0	0	0	2	2
8	1	5	7	17	37	68	135
9	0	0	12	12	37	69	130
10	0	12	50	274	849	—	≥ 1185
11	0	5	41	190	738	—	≥ 974

Table 6. Overview of the number of autocatalytic flows in the formose chemistry. Solutions are grouped by the number of unique reactions used, and by the number of carbon atoms in the largest molecule used. We were not able to compute the missing entries due the demand of computation time (more than 200 hours) and memory (more than 64 GB RAM).

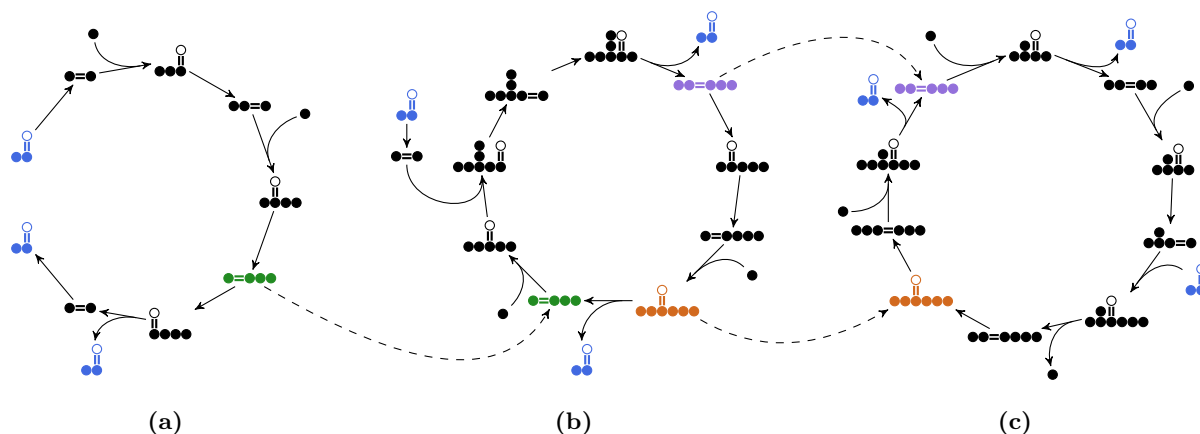


Figure 17. Schematic overview of 3 autocatalytic pathways in the formose chemistry, where carbon-carbon double bonds and carbonyl groups are shown, while hydroxyl groups and hydrogens are implicit. The first pathway can trigger the second pathway via the green molecule, and the second pathway can in turn trigger the third pathway using either the purple or orange molecules. The autocatalytic compound, glycolaldehyde, is shown in blue.

into the metabolic network. The massive drain of material by the feeding of the higher autocatalytic cycles considerably slows down the turn-over of the base cycle or even result in breaking of the cycle. Furthermore, even if the higher autocatalytic cycles themselves would not turn over, the base cycle would still be replenished with C2-C3 compounds generated by retro-aldol fragmentation reactions of longer sugars. This results in a mechanistic scenario where compounds of the base cycle massively fixate formaldehyde in a fast polymerization type process to form longer sugars, which themselves feedback to their starting points on lower levels via fragmentation reactions, refilling these crucial compounds in the base cycle. In that way the fragmentation reactions compensates for the material loss of the autocatalytic base and higher cycles. To better understand the functional aspects of this intricate structure, the couplings between the autocatalytic cycles were analysed. We found, that the interactions between autocatalytic cycles are such, that later cycles in the cascade are triggered by compounds produced by predecessor cycles. Fig. 17 shows an example of such a triggering cascade sequencing 3 cycles, all with the same overall reaction.

6 Discussion

The model presented here, based on integer hyperflows, provides a versatile framework for querying reaction networks for pathways. The restriction to integers, although harder to solve in the general case, has the advantage that the network flow solution can be directly interpreted in terms of mechanisms. Furthermore, questions such as “how many of a specific product can be formed from a limited amount of starting molecules” can easily be formulated and answered in our framework, due to the use of Integer Linear Programming. The ILP approach also enables a targeted enumeration of pathways of interest.

Autocatalysis, for instance, is frequently discussed as one of the key mechanistic concepts to understand the transition from abiotic to biotic chemistry. Reaction chemistries that “maximize” the emergence of this reaction pattern are considered the most plausible predecessors of the chemistries employed by present-day biochemistry. To identify these potential precursor reaction chemistries, a strict algorithmic approach for the search and identification of functional subnetworks in arbitrary reaction chemistry is indispensable. We applied our technique to the formose process and found an intricate topological structure of cascading autocatalytic cycles that feed upon each other, fitting well to the existing experimental evidence. By switching the objective function, different “optimal” solutions can be identified in the same hypergraph as demonstrated with the chemical space of the TCA cycle.

The NOG problem is a prime example of the problem setting in engineering cellular metabolisms, a branch of Synthetic Biology. Given starting material, a target molecule, and a set of enzymes, the task is to find a network that implements the desired chemical transformation. Because of lack of efficient computational network design methods, the established work flow rests on directed evolution and screening [103]. This requires searching for the desired transformation network in metabolic networks of existing organisms, transplanting the identified pathways into the microbial cell factory, followed by improving the performance of the pathway in the alien environment of the host cell. Although impressive examples of this approach have been published [45, 47, 104], this strategy must fail, if no natural pathway, implementing the desired transformation, is known.

Finally, the combination of a graph grammar-based generative chemistry with hyperflow optimisation techniques results in a extremely powerful framework for attacking a wide range of problems from Chemistry and Biology. The grounding of the graph grammar approach in well established mathematical theory allows us to provide

efficient algorithms for an intermediary level of detail. For instance, the Double Pushout formulation of the reaction rules gives us a handle on tracking individual atoms throughout the network, e.g., following specific pathways. The graph transformation formalism also makes it possible to lump reaction sequences into a single overall reaction [84], thereby enabling precise coarse graining operations on reaction networks.

Acknowledgements

This work was supported in part by the Volkswagen Stiftung proj. no. I/82719, and the COST-Action CM1304 “Systems Chemistry” and by the Danish Council for Independent Research, Natural Sciences. It is also supported by the ELSI Origins Network (EON), which is supported by a grant from the John Templeton Foundation. The opinions expressed in this publication are those of the authors and do not necessarily reflect the views of the John Templeton Foundation. We are additionally grateful for feedback on an earlier version of this manuscript provided by Kamelåsa.

A S1 Appendix. Reduction from Independent Set to Maximum Output

This reduction is intended to serve as an example for Sec. 3, in the context of LP relaxation. For a wider set of proofs of the computational complexity of integer hyperflow problems, see [29], where also reductions to bounded hyperedge degree networks are described. We here reduce between the optimisation versions of the problems, though it can easily be adopted to the decision versions. In order to easily compare the reduction to the LP relaxation of the INDEPENDENT SET problem, we will also state the ILP formulation of the problem.

INDEPENDENT SET: Given an undirected graph G , find a set of vertices $V' \subseteq V(G)$, of maximum cardinality, such that no edge in the graph is between vertices of V' , i.e., $E(G) \cap V' \times V' = \emptyset$.

MAXIMUM OUTPUT: Given

- a directed multi-hypergraph \mathcal{H} ,
- a special output vertex $t \in V(\mathcal{H})$,
- and lower and upper bounds on flow $l, u: \overline{E}(\mathcal{H}) \rightarrow \mathbb{N}_0$,

find an integer hyperflow $f: \overline{E}(\mathcal{H}) \rightarrow \mathbb{N}_0$, with maximum output of t , i.e., $\max f(e_t^+)$.

ILP Formulation of Independent Set Let $G = (V, E)$ be the input graph.

$$\begin{aligned} \max \quad & \sum_{v \in V} x_v \\ \text{s.t.} \quad & x_u + x_v \leq 1 && \forall (u, v) \in E \\ & x_v \in \{0, 1\} && \forall v \in V \end{aligned}$$

The resulting independent set is the vertices $v \in V$ with $x_v = 1$.

Reduction to Maximum Output Let G be the input graphs for the INDEPENDENT SET problem. We will first construct a directed multi-hypergraph \mathcal{H} :

$$\begin{aligned}\mathcal{H} &= (V(\mathcal{H}), E(\mathcal{H})) \\ V(\mathcal{H}) &= \{v_g\} \cup \{v_e \mid e \in E(G)\} \\ E(\mathcal{H}) &= \{(\{v_e \mid e \in \delta(v)\}, \{v_g\}) \mid v \in V(G)\}\end{aligned}$$

We thus construct a vertex for each edge in G and an extra ‘‘goal vertex’’. The hyperedges correspond to the vertices of G , with the goal vertex as the head and the rest of the vertices corresponding to the incident edges of G as the tail.

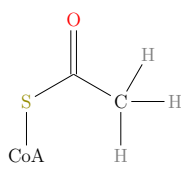
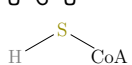
The hypergraph is then I/O-extended to $\bar{\mathcal{H}}$, and we define the lower bound on flow to be 0 on all hyperedges. We set the upper bound to 0 for the input flow to the goal vertex, and 1 for the input to the remaining vertices. All other upper bounds are left infinite.

After maximisation of the output flow of the goal vertex we can construct the independent set as all the vertices $v \in V(G)$ where the corresponding hyperedge $e \in E(\mathcal{H})$ has $f(e) = 1$.

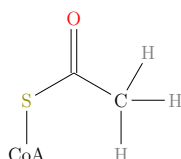
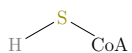
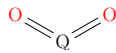
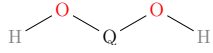
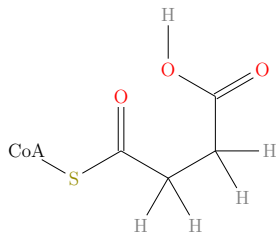
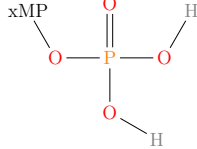
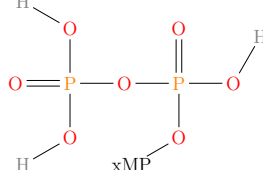
B S2 Appendix. Molecules

The following sections contains tables of the molecule abbreviations used in the main text. Some molecules are modelled using non-chemical vertex labels that represent unimportant substructures. For those molecules we explicitly visualise the labelled graph, while for the strictly chemical molecules we show a SMILES string.

B.1 Molecule Abbreviations Rules for NOG

Abbreviations	Name	SMILES/Visualisation
AcCoA	Acetyl-CoA	
AcP	Acetyl phosphate	<chem>OP(=O)(=O)OC(=O)C</chem>
C ₈ P		<chem>OCC(C(O)C(O)C(O)C(O)COP(=O)(O)=O)=O</chem>
CO ₂	Carbon dioxide	<chem>O=C=O</chem>
CoA	Coenzyme A	
DHAP	Dihydroxyacetone phosphate	<chem>OP(=O)(=O)OCC(=O)CO</chem>
E4P	Erythrose 4-phosphate	<chem>OP(=O)(=O)OCC(O)C(O)C=O</chem>
F6P	Fructose 6-phosphate	<chem>OCC(=O)C(O)C(O)COP(=O)(O)O</chem>
FBP	Fructose 1,6-bisphosphate	<chem>OC(COP(=O)(O)=O)C(O)C(O)C(COP(=O)(O)=O)=O</chem>
G3P	Glyceraldehyde 3-phosphate	<chem>C(C(C=O)O)OP(=O)(O)O</chem>
H ₂ O	Water	<chem>O</chem>
P _i	Phosphate	<chem>O=P(=O)(O)O</chem>
R5P	Ribose 5-phosphate	<chem>OP(=O)(=O)OCC(O)C(O)C(O)C=O</chem>
Ru5P, X5P	Ribulose 5-phosphate, Xylulose 5-phosphate	<chem>OCC(=O)C(O)C(O)COP(=O)(O)O</chem>
S7P	Sedoheptulose 7-phosphate	<chem>O=P(=O)(O)C(C(O)C(O)C(O)C(=O)CO)O</chem>
SBP	Sedoheptulose 1,7-bisphosphate	<chem>OC(COP(=O)(O)=O)C(O)C(O)C(O)C(COP(=O)(O)=O)=O</chem>

B.2 Molecule Abbreviations for TCA and Glyoxylate

Abbreviations	Name	SMILES/Visualisation
AcCoA	Acetyl-CoA	 <chem>CC(=O)SCoA</chem>
Acon	Aconitate	<chem>O=C(O)CC(=CC(=O)O)C(=O)O</chem>
AKG	2-oxoglutarate	<chem>O=C(O)C(=O)CCC(=O)O</chem>
Cit	Citrate	<chem>C(C(=O)O)C(CC(=O)O)(C(=O)O)O</chem>
CO ₂	Carbon dioxide	<chem>O=C=O</chem>
CoA	Coenzyme A	 <chem>SCoA</chem>
Fum	Fumarate	<chem>C(=CC(=O)O)C(=O)O</chem>
GLX	Glyoxylate	<chem>C(=O)C(=O)O</chem>
H ⁺	Proton	<chem>[H+]</chem>
H ₂ O	Water	<chem>O</chem>
ICit	Isocitrate	<chem>O=C(O)C(CC(=O)O)C(O)C(=O)O</chem>
Mal	Malate	<chem>O=C(O)CC(O)C(=O)O</chem>
NAD ⁺	Oxidized nicotinamide adenine dinucleotide	<chem>NAD+</chem>
NADH	Reduced nicotinamide adenine dinucleotide	<chem>H-NAD</chem>
OAA	Oxaloacetate	<chem>O=C(O)C(=O)CC(=O)O</chem>
OASuc	Oxalosuccinate	<chem>C(C(C(=O)C(=O)O)C(=O)O)C(=O)O</chem>
P _i	Phosphate	<chem>O=P(O)(O)O</chem>
Pyr	Pyruvate	<chem>C(CC(=O)O)C(=O)O</chem>
Q	Quinone	 <chem>O=C1C=CC(=O)C=C1</chem>
QH ₂	Hydroquinone	 <chem>O=C1C=CC(=O)C=C1</chem>
Suc	Succinate	<chem>C(CC(=O)O)C(=O)O</chem>
SucCoA	Succinyl-CoA	 <chem>CC(=O)SCoA</chem>
xDP		 <chem>O=P(O)(O)O</chem>
xTP		 <chem>O=P(O)(O)OP(=O)(O)O</chem>

B.3 Molecule Abbreviations for Formose

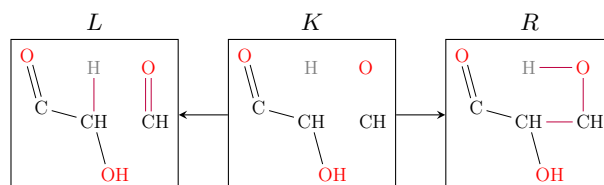
Abbreviations	Name	SMILES/Visualisation
C ₁	Formaldehyde	C=O
C _{2a}	Glycolaldehyde	OCC=O
C _{2e}		OC=CO
C _{3a}		OCC(O)C=O

C S3 Appendix. Transformation Rules

The following sections contains visualisations of all graph transformation rules used to generated the analysed networks. Each rule is annotated with references to the data its modelling is based on. Most of these reference are to entries in the MACiE (Mechanism, Annotation, and Classification in Enzymes) database [105,106].

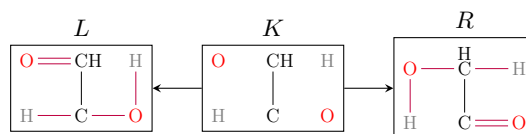
C.1 Transformation Rules for NOG

C.1.1 Aldolase



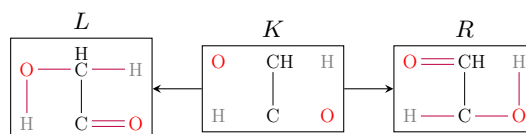
MACiE entry 0052.

C.1.2 Aldose-Ketose



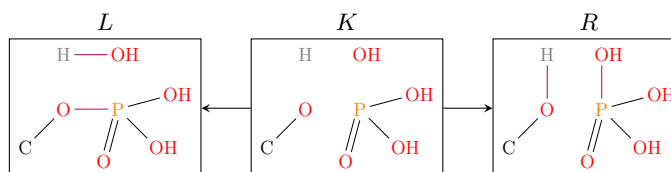
MACiE entry 0308.

C.1.3 Ketose-Aldose



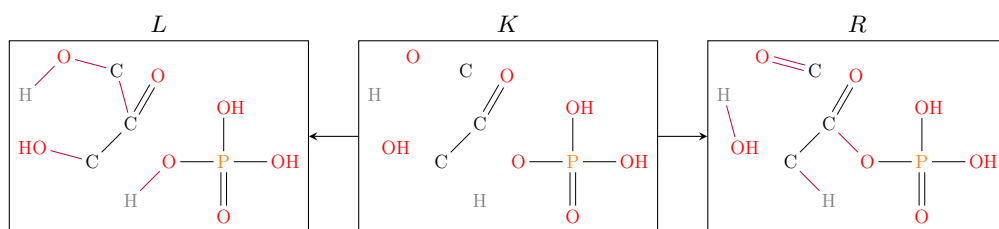
MACiE entry 0308.

C.1.4 Phosphohydrolase



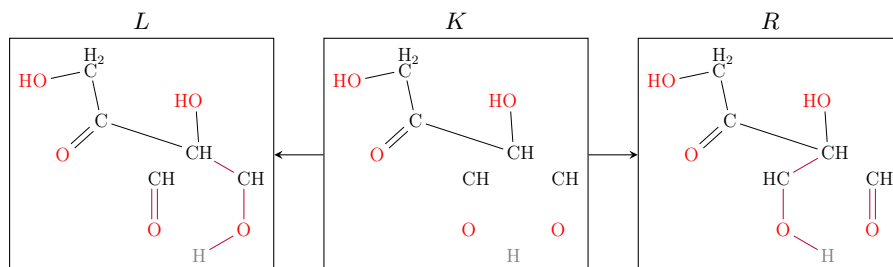
MACiE entries 0043, 0044, and 0047.

C.1.5 Phosphoketolase



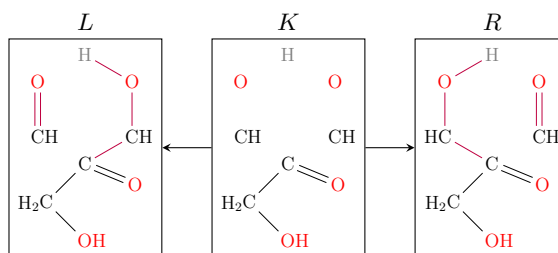
MetaCyc [107] reaction entry 4.1.2.22.

C.1.6 Transaldolase



MACiE entry 0148.

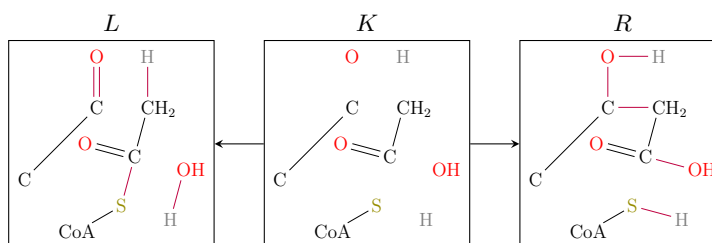
C.1.7 Transketolase



MACiE entry 0219.

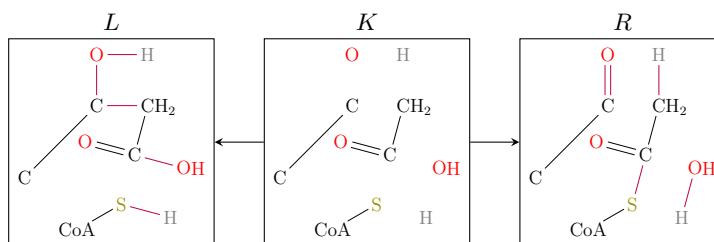
C.2 Transformation Rules for TCA and Glyoxylate

C.2.1 AcCoACarbonylTrans



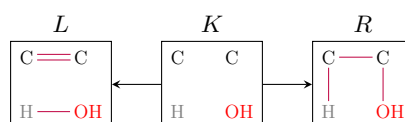
MACiE entries 0053 and 0078. The vertex label CoA is used as a shorthand for representing all of coenzyme A, except for the terminating SH group.

C.2.2 AcCoACarbonylTrans, inverse



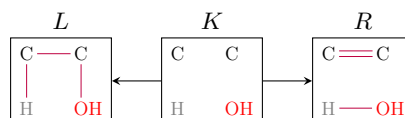
MACiE entries 0053 and 0078. The vertex label CoA is used as a shorthand for representing all of coenzyme A, except for the terminating SH group.

C.2.3 Hydration



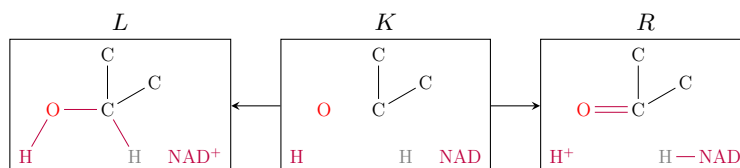
MACiE entries for EC 4.1.2.*.

C.2.4 Dehydration



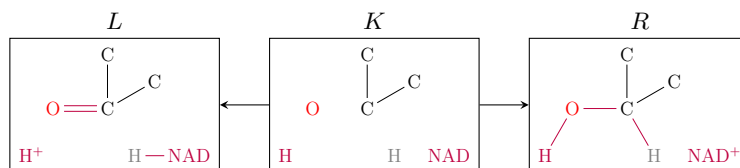
MACiE entries for EC 4.1.2.*.

C.2.5 NAD Oxidoreductase



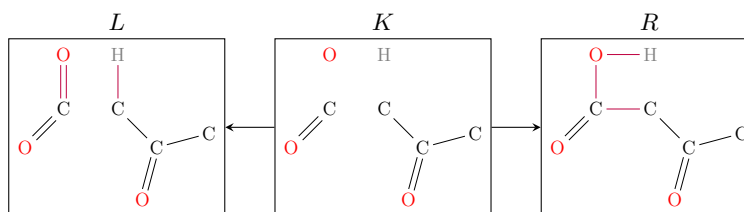
MACiE entries 0007 and 0021. The vertex labels NAD+ and NAD are used to represent an entire NAD molecule. Though, the reduced form has an attached hydrogen atom.

C.2.6 NAD Oxidoreductase, inverse



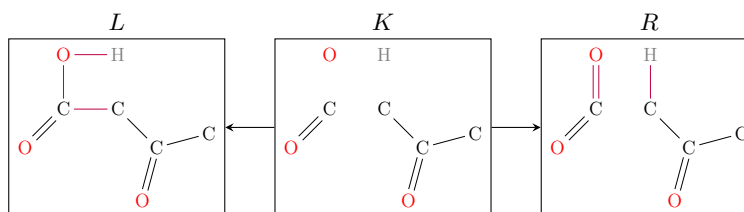
MACiE entries 0007 and 0021. The vertex labels NAD+ and NAD are used to represent an entire NAD molecule. Though, the reduced form has an attached hydrogen atom.

C.2.7 Carboxylation



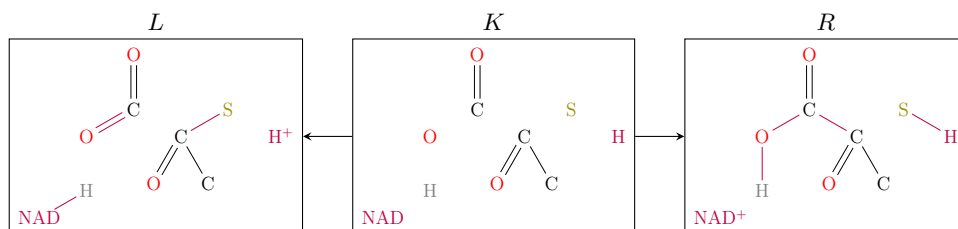
MACiE entries 0007 and 0021.

C.2.8 Decarboxylation



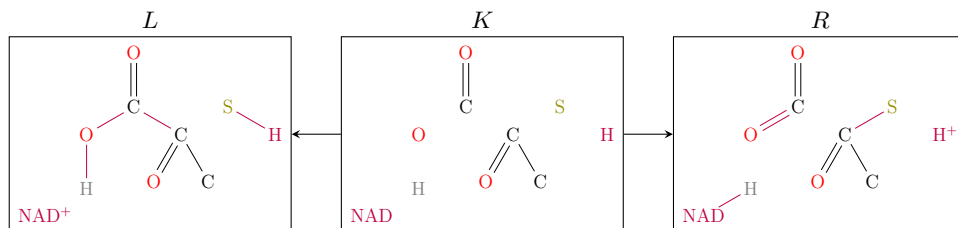
MACiE entries 0007 and 0021.

C.2.9 Carboxylation, S-trans



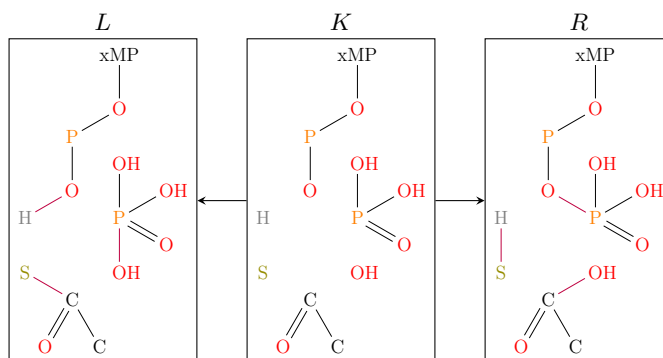
MACiE entries 0106 and 0280. The vertex labels NAD⁺ and NAD are used to represent an entire NAD molecule. Though, the reduced form has an attached hydrogen atom.

C.2.10 Decarboxylation, S-trans



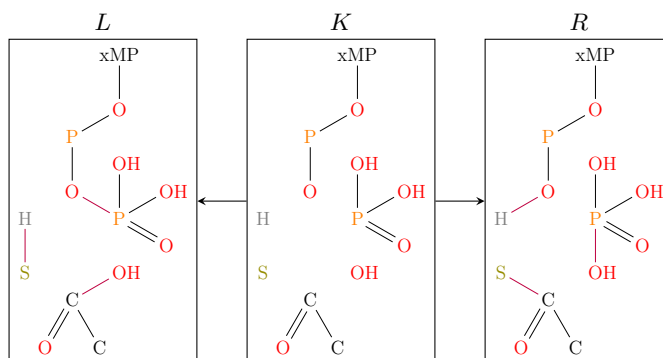
MACiE entries 0106 and 0280. The vertex labels NAD⁺ and NAD are used to represent an entire NAD molecule. Though, the reduced form has an attached hydrogen atom.

C.2.11 Acid CoA Ligase



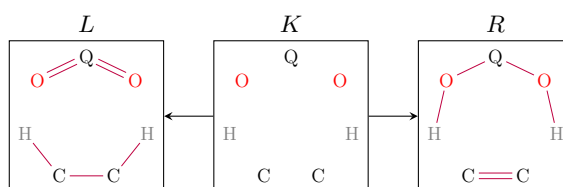
MACiE entry 0198. The vertex label xMP is used as a shorthand for representing any of the nucleoside monophosphates, e.g., AMP.

C.2.12 Acid CoA Ligase, inverse



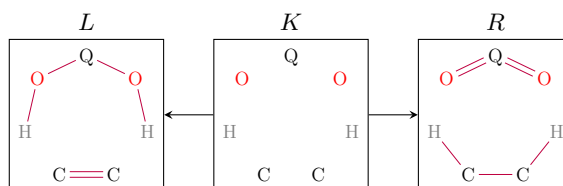
MACiE entry 0198. The vertex label xMP is used as a shorthand for representing any of the nucleoside monophosphates, e.g., AMP.

C.2.13 Q Dehydrogenase



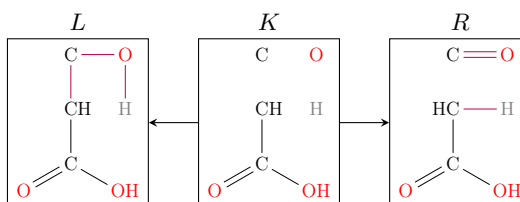
MACiE entry 0294. The vertex label Q is used to model the cycle in both hydroquinone and quinone.

C.2.14 Q Dehydrogenase, inverse



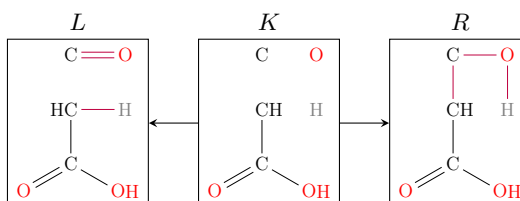
MACiE entry 0294. The vertex label Q is used to model the cycle in both hydroquinone and quinone.

C.2.15 Oxo Acid Lyase



MACiE entry 0272.

C.2.16 Oxo Acid Lyase, inverse

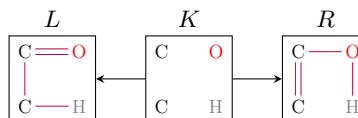


MACiE entry 0272.

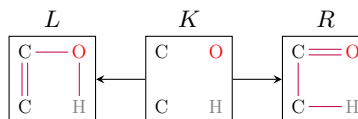
C.3 Transformation Rules for Formose

The two reversible reactions have been modelled based on [100].

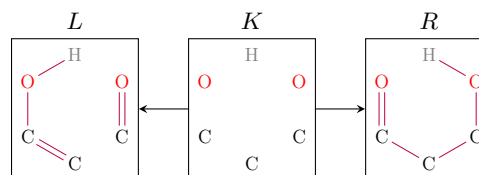
C.3.1 Keto-Enol



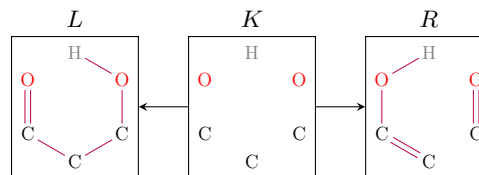
C.3.2 Enol-Keto



C.3.3 Aldol addition



C.3.4 Retro-aldol addition



References

1. Papoutsakis ET. Equations and calculations for fermentations of butyric acid bacteria. *Biotech Bioeng.* 1984;26:174–187.
2. Watson MR. Metabolic maps for the Apple II. *Biochem Soc Trans.* 1984;12:1093–1094.
3. Fell DA, Small JR. Fat synthesis in adipose tissue. An examination of stoichiometric constraints. *Biochem J.* 1986;238:781–786.
4. Kauffman KJ, Prakash P, Edwards JS. Advances in flux balance analysis. *Current Opinion Biotech.* 2003;14:491–496.
5. Orth JD, Thiele I, Palsson BØ. What is flux balance analysis? *Nat Biotech.* 2010;28:245–248.
6. Schuster S, Hilgetag C. On elementary flux modes in biochemical reaction systems at steady state. *J Biol Syst.* 1994;2:165–182.
7. Schuster S, Fell DA, Dandekar T. A general definition of metabolic pathways useful for systematic organization and analysis of complex metabolic networks. *Nat Biotechnol.* 2000;18:326–332.
8. Behre J, de Figueiredo LF, Schuster S, Kaleta C. Detecting structural invariants in biological reaction networks. *Methods Mol Biol.* 2012;804:377–407.
9. Zanghellini J, Ruckerbauer DE, Hanscho M, Jungreuthmayer C. Elementary flux modes in a nutshell: properties, calculation and applications. *Biotechnol J.* 2013;8:1009–1016.
10. Klamt S, Stelling J. Combinatorial complexity of pathway analysis in metabolic networks. *Mol Biol Rep.* 2002;29:233–236.
11. Klamt S, Stelling J. Two approaches for metabolic pathway analysis? *Trends Biotechnol.* 2003;21:64–69.
12. Wagner C, Urbanczik R. The geometry of the flux cone of a metabolic network. *Biophys J.* 2005;89:3837–3845.
13. Kaleta C, Centler F, Dittrich P. Analyzing molecular reaction networks: from pathways to chemical organizations. *Mol Biotechnol.* 2006;34:117–123.
14. Centler F, Kaleta C, Speroni di Fenizio P, Dittrich P. Computing chemical organizations in biological networks. *Bioinformatics.* 2008;24:1611–1618.
15. Kaleta C, Richter S, Dittrich P. Using chemical organization theory for model checking. *Bioinformatics.* 2009;25:1915–1922.
16. Planes FJ, Beasley JE. Path finding approaches and metabolic pathways. *Discrete Applied Mathematics.* 2009;157(10):2244 – 2256. *Networks in Computational Biology.*
17. Planes FJ, Beasley JE. A critical examination of stoichiometric and path-finding approaches to metabolic pathways. *Briefings in Bioinformatics.* 2008;9(5):422–436.

-
18. Hucka M, Finney A, Bornstein BJ, Keating SM, Shapiro BE, Matthews J, et al. Evolving a lingua franca and associated software infrastructure for computational systems biology: the Systems Biology Markup Language (SBML) project. *Systems biology*. 2004;1(1):41–53.
 19. Kanehisa M, Goto S, Sato Y, Furumichi M, Tanabe M. KEGG for integration and interpretation of large-scale molecular datasets. *Nucleic Acids Res*. 2012;40:D109–D114.
 20. Altman T, Travers M, Kothari A, Caspi R, Karp PD. A systematic comparison of the MetaCyc and KEGG pathway databases. *BMC Bioinformatics*. 2013;14:112.
 21. Flamm C, Ullrich A, Ekker H, Mann M, Högerl D, Rohrschneider M, et al. Evolution of Metabolic Networks: A Computational Framework. *J Syst Chem*. 2010;1:4.
 22. Andersen JL, Flamm C, Merkle D, Stadler PF. Inferring Chemical Reaction Patterns Using Graph Grammar Rule Composition. *J Syst Chem*. 2013;4:4.
 23. Andersen JL, Flamm C, Merkle D, Stadler PF. Generic Strategies for Chemical Space Exploration. *International Journal of Computational Biology and Drug Design*. 2014;7(2/3):225 – 258.
 24. Beasley JE, Planes FJ. Recovering metabolic pathways via optimization. *Bioinformatics*. 2007;23(1):92–98.
 25. Pistorius J, Minoux M. An improved direct labeling method for the max-flow min-cut computation in large hypergraphs and applications. *Intl Trans in Op Res*. 2003;10:1–11.
 26. Ahuja RK, Magnanti TL, Orlin JB. *Network Flows: Theory, Algorithms, and Applications*. Englewood Cliffs, NJ: Prentice Hall; 1993.
 27. Cambini R, Gallo G, Scutellà MG. Flows on hypergraphs. *Mathematical Programming*. 1997;78:195–217.
 28. Gallo G, Gentile C, Pretolani D, Rago G. Max Horn SAT and the minimum cut problem in directed hypergraphs. *Math Programming*. 1998;80:213–237.
 29. Andersen JL, Flamm C, Merkle D, Stadler PF. Maximizing output and recognizing autocatalysis in chemical reaction networks is NP-complete. *J Systems Chem*. 2012;3:1.
 30. Karp RM. Reducibility among combinatorial problems. In: Miller RE, Thatcher JW, editors. *Complexity of Computer Computations*. NY: Plenum Press; 1972. p. 85–103.
 31. Balaban AT. *Chemical applications of graph theory*. London: Academic Press; 1976.
 32. Trinajstić N. *Chemical graph theory*. Boca Raton, FL: CRC Press; 1983.
 33. Bonchev D, Rouvray DH, editors. *Chemical graph theory: reactivity and kinetics*. Philadelphia, PA: Gordon and Breach; 1992.
 34. Tomlin AS, Turányi T, Pilling MJ. Mathematical Tools for the Construction, Investigation and Reduction of Combustion Mechanisms. In: *Comprehensive Chemical Kinetics*. vol. 35. Amsterdam: Elsevier; 1997. p. 293–437.

-
35. Benkő G, Flamm C, Stadler PF. A graph-based toy model of chemistry. *J Chem Inf Comput Sci.* 2003;43(4):1085–1093.
 36. Yadav MK, Kelley BP, Silverman SM. The Potential of a Chemical Graph Transformation System. In: Ehrig H, Engels G, Parisi-Presicce F, Rozenberg G, editors. *Graph Transformations.* vol. 3256 of *Lect. Notes Comp. Sci.* Berlin, Heidelberg: Springer; 2004. p. 83–95.
 37. Rosselló F, Valiente G. Chemical Graphs, Chemical Reaction Graphs, and Chemical Graph Transformation. *Electronic Notes Theor Comp Sci.* 2005;127:157–166.
 38. Bournez O, Ibăescu L, Kirchner H. From Chemical Rules to Term Rewriting. *Electronic Notes Theor Comp Sci.* 2006;147:113–134.
 39. Blinov ML, Yang J, Faeder JR, Hlavacek WS. Graph theory for rule-based modeling of biochemical networks. In: Priami C, Ingolfsdottir A, Mishra B, Nielson H, editors. *Transactions on Computational Systems Biology VII.* vol. 4230 of *Lect. Notes Comp. Sci.* Berlin, Heidelberg: Springer; 2006. p. 89–106.
 40. Ehrig H, Ehrig K, Prange U, Taentzner G. *Fundamentals of Algebraic Graph Transformation.* Berlin, D: Springer-Verlag; 2006.
 41. Grzybowski BA, Bishop KJM, Kowalczyk B, Wilmer CE. The 'wired' universe of organic chemistry. *Nature Chemistry.* 2009;1:31–36.
 42. Meléndez-Hevia E, Isidoro A. The game of the pentose phosphate cycle. *J Theor Biol.* 1985;117:251–263.
 43. Kleijn RJ, van Winden WA, van Gulik WM, Heijnen JJ. Revisiting the ¹³C-label distribution of the non-oxidative branch of pentose phosphate pathway based upon kinetic and genetic evidence. *FEBS Journal.* 2005;272:4970–4982.
 44. Bogorad IW, Lin TS, Liao JC. Synthetic non-oxidative glycolysis enables complete carbon conservation. *Nature.* 2013;502(7473):693–697.
 45. Thodey K, Galanie S, Smolke CD. A microbial biomanufacturing platform for natural and semisynthetic opioids. *Nature Chem Biol.* 2014;10:837–844.
 46. Nielsen J. Yeast cell factories on the horizon. *Science.* 2015;349:1050–1051.
 47. Ro DK, Paradise EM, Ouellet M, Fisher KJ, Newman KL, Ndungu JM, et al. Production of the antimalarial drug precursor artemisinic acid in engineered yeast. *Nature.* 2006;440:940–943.
 48. DeLoache WC, Russ ZN, Narcross L, Gonzales AM, Martin VJJ, Dueber JE. An enzyme-coupled biosensor enables (S)-reticuline production in yeast from glucose. *Nature Chem Biol.* 2015;11:465–471.
 49. Rappoport D, Galvin CJ, Zubarev DY, Aspuru-Guzik A. Complex Chemical Reaction Networks from Heuristics-Aided Quantum Chemistry. *J Chem Theory Comput.* 2014;10:897–907.
 50. Ruiz-Mirazo K, Briones C, de la Escosura A. Prebiotic Systems Chemistry: New Perspectives for the Origins of Life. *Chem Rev.* 2014;114:285–366.
 51. Bang-Jensen J, Gutin GZ. *Digraphs: Theory, Algorithms and Applications.* Springer Monographs in Mathematics. 2009;.

-
52. Jeroslow RG, Martin RK, Rardin RR, Wang J. Gainfree Leontief substitution flow problems. *Mathematical Programming*. 1992;57:375–414.
 53. Bissette AJ, Fletcher SP. Mechanisms of Autocatalysis. *J Angew Chemie Int Ed*. 2013;52(49):12800–12826.
 54. Eigen M, Schuster P. The Hypercycle. A Principle of Natural Self-Organisation. Part A: Emergence of the Hypercycle. *Naturwissenschaften*. 1977;64:541–565.
 55. Eigen M, Schuster P. The Hypercycle. A Principle of Natural Self-Organisation. Part B: The Abstract Hypercycle. *Naturwissenschaften*. 1979;65:7–41.
 56. Eigen M, Schuster P. The Hypercycle. A Principle of Natural Self-Organisation. Part C: The Realistic Hypercycle. *Naturwissenschaften*. 1979;65:341–369.
 57. Handorf T, Ebenhöf O, Heinrich R. Expanding Metabolic Networks: Scopes of Compounds, Robustness, and Evolution. *Journal of Molecular Evolution*. 2005;61:498–512. 10.1007/s00239-005-0027-1.
 58. Ebenhöf O, Handorf T, Heinrich R. Structural analysis of expanding metabolic networks. In: *Genome informatics. International Conference on Genome Informatics*. vol. 15; 2004. p. 35.
 59. Kun Á, Papp B, Szathmáry E. Computational identification of obligatorily autocatalytic replicators embedded in metabolic networks. *Genome Biol*. 2008;9:R51.
 60. Ebenhöf O, Heinrich R. Stoichiometric design of metabolic networks: multifunctionality, clusters, optimization, weak and strong robustness. *Bull Math Biol*. 2003;65:323–357.
 61. Meléndez-Hevia TG Waddell, Montero F. Optimization of Metabolism: The Evolution of Metabolic Pathways Towards Simplicity Through the Game of the Pentose Phosphate Cycle. *J Theor Biol*. 1994;166:201–220.
 62. Noor E, Eden E, Milo R, Alon U. Central Carbon Metabolism as a Minimal Biochemical Walk between Precursors for Biomass and Energy. *Mol Cell*. 2010;39:809–820.
 63. Bar-Even A, Noor E, Lewis NE, Milo R. Design and analysis of synthetic carbon fixation pathways. *Proc Natl Acad Sci*. 2010;107:8889–8894.
 64. Papin JA, Stelling J, Price ND, Klamt S, Schuster S, Palsson BO. Comparison of network-based pathway analysis methods. *Trends in biotechnology*. 2004;22(8):400–405.
 65. García-Junceda E, Lavandera I, Rother D, Schittwieser JH. (Chemo)enzymatic cascades – Nature’s synthetic strategy transferred to the laboratory. *J Mol Cat B: Enzymatic*. 2015;114:1–6.
 66. Choi JM, Han SS, Kim HS. Industrial applications of enzyme biocatalysis: Current status and future aspects. *Biotech Adv*. 2015;33:1443–1454.
 67. Ricca E, Brucher B, Schittwieser JH. Multi-Enzymatic Cascade Reactions: Overview and Perspectives. *Adv Synth Catal* 353, 2239 – 2262. 2011;353:2239–2262.
 68. Schuster S, Hilgetag C. On elementary flux modes in biochemical reaction systems at steady state. *Journal of Biological Systems*. 1994;2(02):165–182.

-
69. Schilling CH, Letscher D, Palsson BØ. Theory for the systemic definition of metabolic pathways and their use in interpreting metabolic function from a pathway-oriented perspective. *Journal of theoretical biology.* 2000;203(3):229–248.
 70. Savinell JM, Palsson BØ. Network Analysis of Intermediary Metabolism using Linear Optimization. I. Development of Mathematical Formalism. *J theor Biol.* 1992;154:421–454.
 71. Khachiyan LG. Polynomial algorithms in linear programming. *USSR Computational Mathematics and Mathematical Physics.* 1980;20(1):53–72.
 72. Karmarkar N. A new polynomial-time algorithm for linear programming. In: *Proceedings of the sixteenth annual ACM symposium on Theory of computing.* ACM; 1984. p. 302–311.
 73. Guptasarma P. Does replication-induced transcription regulate synthesis of the myriad low copy number proteins of *Escherichia coli*? *Bioessays.* 1995;17(11):987–997.
 74. Fedoroff N, Fontana W. Small Numbers of Big Molecules. *Science.* 2002;297:1129–1130.
 75. Milo R, Phillips R. *Cell Biology by the Numbers.* New York, NY: Garland Science; 2015.
 76. Gillespie DT. Stochastic simulation of chemical kinetics. *Annu Rev Phys Chem.* 2007;58:35–55.
 77. Érdi P, Lente G. *Stochastic Chemical Kinetics: Theory and (Mostly) Systems Biological Applications.* New York: Springer; 2014.
 78. Kreyszig P, Wozar C, Peter S, Veloz T, Ibrahim B, Dittrich P. Effects of small particle numbers on long-term behaviour in discrete biochemical systems. *Bioinformatics.* 2014;30:i475–i481.
 79. Matheiss T, Rubin DS. A survey and comparison of methods for finding all vertices of convex polyhedral sets. *Mathematics of operations research.* 1980;5(2):167–185.
 80. Swart G. Finding the convex hull facet by facet. *Journal of Algorithms.* 1985;6(1):17–48.
 81. Lee S, Phalakornkule C, Domach MM, Grossmann IE. Recursive MILP model for finding all the alternate optima in LP models for metabolic networks. *Computers & Chemical Engineering.* 2000;24(2):711–716.
 82. Andersen JL, Flamm C, Merkle D, Stadler PF. A Software Package for Chemically Inspired Graph Transformation; 2016. Submitted, TR: <http://arxiv.org/abs/1603.02481>.
 83. Andersen JL. MedØIDatschgerl (MØD); 2016. <http://mod.imada.sdu.dk>.
 84. Andersen JL, Flamm C, Merkle D, Stadler PF. 50 Shades of Rule Composition: From Chemical Reactions to Higher Levels of Abstraction. In: Fages F, Carla Piazza C, editors. *Proceedings of the 1st International Conference on Formal Methods in Macro-Biology.* vol. 8738 of LNCS. Springer-Verlag, Berlin Heidelberg; 2014. p. 117–135.

-
85. Andersen JL, Flamm C, Hanczyc MM, Merkle D. Towards an Optimal DNA-Templated Molecular Assembler. In: ALIFE 14: The Fourteenth Conference on the Synthesis and Simulation of Living Systems. vol. 14; 2014. p. 557–564.
 86. Andersen JL, Andersen T, Flamm C, Hanczyc MM, Merkle D, Stadler PF. Navigating the Chemical Space of HCN Polymerization and Hydrolysis: Guiding Graph Grammars by Mass Spectrometry Data. *Entropy*. 2013;15(10):4066–4083. Available from: <http://www.mdpi.com/1099-4300/15/10/4066>.
 87. Andersen JL, Flamm C, Merkle D, Stadler PF. In silico Support for Eschenmoser's Glyoxylate Scenario. *Israel Journal of Chemistry*. 2015;.
 88. Stincone A, Prigione A, Cramer T, Wamelink MMC, Campbell K, Cheung E, et al. The return of metabolism: biochemistry and physiology of the pentose phosphate pathway. *Biol Rev*. 2015;90:927–963.
 89. Huynen MA, Dandekar T, Bork P. Variation and evolution of the citric-acid cycle: a genomic perspective. *Trends in Microbiology*. 1999;7(7):281–291.
 90. Liu F, Thatcher JD, Barral JM, Epstein HF. Bifunctional Glyoxylate Cycle Protein of *Caenorhabditis elegans*: A Developmentally Regulated Protein of Intestine and Muscle. *Develop Biol*. 1995;169(2):399–414.
 91. de Figueiredo LF, Schuster S, Kaleta C, Fell DA. Can sugars be produced from fatty acids? A test case for pathway analysis tools. *Bioinformatics*. 2008;24(22):2615–2621.
 92. Fuchs G. Alternative Pathways of Carbon Dioxide Fixation: Insights into the Early Evolution of Life? *Annu Rev Microbiol*. 2011;65:631–658.
 93. Hügler M, Sievert SM. Beyond the Calvin Cycle: Autotrophic Carbon Fixation in the Ocean. *Annu Rev Microbiol*. 2011;3:261–289.
 94. Wächtershäuser G. Before enzymes and templates: theory of surface metabolism. *Microbiol Rev*. 1988;52:452–484.
 95. Morowitz HJ, Kostelnik JD, Yang J, Cody GD. The origin of intermediary metabolism. *Proc Nat Acad Sci*. 2000;97:7704–7708.
 96. Smith E, Morowitz HJ. Universality in intermediary metabolism. *Proc Nat Acad Sci*. 2004;101:13168–13173.
 97. Orgel LE. The implausibility of metabolic cycles on the prebiotic Earth. *PLoS biology*. 2008;6(1):e18.
 98. Butlerov AM. Einiges über die chemische Structur der Körper. *Zeitschrift für Chemie*. 1861;4:549–560.
 99. Delidovich IV, Simonov AN, Taran OP, Parmon VN. Catalytic Formation of Monosaccharides: From the Formose Reaction towards Selective Synthesis. *ChemSusChem*. 2014;7(7):1833–1846.
 100. Benner SA, Kim HJ, Kim MJ, Ricardo A. Planetary Organic Chemistry and the Origins of Biomolecules. *Cold Spring Harbor Perspectives in Biology*. 2010;2(7).
 101. Ricardo A, Frye F, Carrigan MA, Tipton JD, Powell DH, Benner SA. 2-Hydroxymethylboronate as a Reagent To Detect Carbohydrates: Application to the Analysis of the Formose Reaction. *J Org Chem*. 2006;71:9503–9505.

-
102. Kim HJ, Ricardo A, Illangkoon HI, Kim MJ, Carrigan MA, Frye F, et al. Synthesis of Carbohydrates in Mineral-Guided Prebiotic Cycles. *J Am Chem Soc.* 2011;133(24):9457–9468.
 103. Boyle PM, Silver PA. Parts plus pipes: Synthetic biology approaches to metabolic engineering. *Metab Eng.* 2012;14:223–232.
 104. Fossati E, Narcross L, Ekins A, Falguyret JP, Martin VJJ. Synthesis of Morphinan Alkaloids in *Saccharomyces cerevisiae*. *PLoS ONE.* 2015;10(4):e0124459.
 105. Holliday GL, Bartlett GJ, Almonacid DE, O’Boyle NM, Murray-Rust P, Thornton JM, et al. MACiE: a database of enzyme reaction mechanisms. *Bioinformatics.* 2005;21:4315–4316.
 106. Holliday GL, Andreini C, Fischer JD, Rahman SA, Almonacid DE, Williams ST, et al. MACiE: exploring the diversity of biochemical reactions. *Nucleic Acids Research.* 2012;40:D783–D789.
 107. Caspi R, Altman T, Dreher K, Fulcher CA, Subhraveti P, Keseler IM, et al. The **MetaCyc** database of metabolic pathways and enzymes and the **BioCyc** collection of pathway/genome databases. *Nucleic Acids Research.* 2012;40:D742–D753.