Chemical Transformation Motifs—Modelling Pathways as Integer Hyperflows

Jakob L. Andersen, Christoph Flamm, Daniel Merkle, and Peter F. Stadler

Abstract—We present an elaborate framework for formally modelling pathways in chemical reaction networks on a mechanistic level. Networks are modelled mathematically as directed multi-hypergraphs, with vertices corresponding to molecules and hyperedges to reactions. Pathways are modelled as integer hyperflows and we expand the network model by detailed routing constraints. In contrast to the more traditional approaches like Flux Balance Analysis or Elementary Mode analysis we insist on integer-valued flows. While this choice makes it necessary to solve possibly hard integer linear programs, it has the advantage that more detailed mechanistic questions can be formulated. It is thus possible to query networks for general transformation motifs, and to automatically enumerate optimal and near-optimal pathways. Similarities and differences between our work and traditional approaches in metabolic network analysis are discussed in detail. 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 manually designed pathways can be further optimized. We then use a model of sugar chemistry to investigate pathways in the autocatalytic formose process. A graph transformation-based approach is used to automatically generate the reaction networks of interest.

Index Terms—Pathway, hyperflow, network flow, reaction network, integer linear programming, chemistry, autocatalysis, non-oxidative glycolysis pathway, formose

1 Introduction

Chemical reactions are intrinsically many-to-many relationships on molecules. Chemical reaction networks thus are naturally modelled as directed multi-hypergraphs [1], with vertices being molecules and directed hyperedges being reactions. A wide variety of methods has been devised to characterise the function of the networks in terms of pathways. Nevertheless, the very notion of a *pathway* has remained difficult to define formally. While biochemical tradition stipulates that certain combinations of enzyme-catalysed reactions constitute pathways, competing mathematical approaches single out reaction sets with specific properties that may serve as natural pathways in a particular setting. Flux Balance Analysis (FBA) [2], [3], [4] is geared towards finding a single pathway that is optimal in a certain sense, usually w.r.t. a biomass function defined from experiments. In contrast, Elementary Flux Modes (EFM) [5] and Extremal Pathways (ExPa) [6], [7] consider particular sets of pathways that implicitly describe the complete solution space. An important formal difference is that EFMs and ExPas have a mechanistic interpretation as integer valued fluxes, while FBA pathways do not have this property. Reversible reactions are in ExPa represented as separate pairs of mutually inverse hyperedges, while this is not done in FBA and EFM which instead allow for negative fluxes. A general mechanistic model related to EFM pathways introduces additional constraints to limit futile flux [8]. Interpretations of chemical networks that disregard the stoichiometry of the network have been popular since they are amenable to efficient methods from network analysis (e.g., see [9]) but are much less expressive [10].

We introduce here a versatile model for general mechanistic pathways, where routing constraints can be introduced to limit futile branches. Pathways are modelled as *integer*-valued hyperflows, which are the natural generalization of flows on directed graphs to directed hypergraphs [11]. Hyperflows are mathematically equivalent to chemical fluxes. They have been studied in their own right for restricted classes of hypergraphs [12], [13] and provide a direct link to the rich computer science literature (see e.g., [14]). Hyperflows can be found with Linear Programming (LP), as in FBA. We will show, however, that there are cases where it is important to insist on integer hyperflows. Integer hyperflows can be found with Integer Linear Programming (ILP), a technique that has been applied in previous pathway models [8], [10] and that we will also make extensive use of. As we shall see, LP and ILP solutions can be vastly different. It is well-known that LP can be solved in

- *J.L. Andersen is with the Earth-Life Science Institute, Tokyo Institute of Technology, Tokyo 152-8550, Japan, the Institute for Theoretical Chemistry, University of Vienna, Wien A-1090, Austria, and the Department of Mathematics and Computer Science, University of Southern Denmark, Odense M DK-5230, Denmark. E-mail: jakob@caput.dk.*
- *C. Flamm is with the Institute for Theoretical Chemistry, University of Vienna, Wien A-1090, Austria. E-mail: xtof@tbi.univie.ac.at.*
- *D. Merkle is with the Department of Mathematics and Computer Science, University of Southern Denmark, Odense M DK-5230, Denmark. E-mail: daniel@imada.sdu.dk.*
- *P.F. Stadler is with the Institute for Theoretical Chemistry, University of Vienna, Wien A-1090, Austria, and the Bioinformatics Group, Department of Computer Science, and Interdisciplinary Center for Bioinformatics, University of Leipzig, Leipzig D-04107, Germany. E-mail: studla@bioinf.uni-leipzig.de.*

Manuscript received 14 Mar. 2017; revised 19 Sept. 2017; accepted 17 Nov. 2017. Date of publication 11 Dec. 2017; date of current version 29 Mar. 2019. (Corresponding author: Daniel Merkle.)
For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below.
Digital Object Identifier no. 10.1109/TCBB.2017.2781724

polynomial time [15], [16], while ILP is NP-hard in the general case [17]. Not surprisingly, the restricted problem of finding a maximum integer hyperflow in a chemical reaction network is also NP-hard, even for bi-molecular reactions (bounded degree hyperedges) [18]. Nevertheless, modern ILP solvers readily deal with most problem instances of practical relevance. Specialised (Mixed) ILP modification schemes have been devised to enumerate EFMs [19] and minimal reaction sets in an FBA setting [20], [21], [22]. Here we use a tree search algorithm in combination with the underlying ILP solver to enumerate alternative optimal and near-optimal pathways.

Pathways are only one way to capture the structure of a chemical network. The theory of Chemical Organizations (CO) [23], [24] considers higher-level properties of pathways and focusses on closure properties of subnetworks. A closely related topic is the recognition of catalytic and autocatalytic cycles. Autocatalysis forms the basis of autopoietic systems, a particular type of organizational structure. In prebiotic chemistry [25], autocatalysis has long been hypothesized to be one of the dominating mechanisms that eventually lead up to the origin of life. Logically described by the simple scheme $(A) + X \to n\,X + (B)$, autocatalysis may be instantiated by a wide variety of distributed chemical schemes [26], [27] akin to autocatalytic set models [28], [29]. We will show that the geometric interpretation of integer hyperflows also facilitates the definition of algebraic criteria for autocatalyis, thereby allowing us to identify autocatalytic cycles as the solutions of an ILP problem. To this end we devise here an expanded network model that represents each vertex (compound) as a bipartite digraph. This allows us to efficiently implement additional topological constraints to handle futile cycles and to treat catalytic and autocatalytic pathways that are not immediately meaningful in the traditional FBA framework without mechanistic pathways.

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 with a well-defined interface to the remainder of the network and implements a coherent overall chemical transformation. Metabolic pathways and subnetworks exhibiting overall (auto) catalysis are particular examples of CTMs. Hence CTMs are formal specification of a set of integer hyperflows in a chemical reaction network, possibly in conjunction with additional constraints on the allowed hyperflows. As an example we will investigate the many ways of instantiating the transformation motif *defined* by the conversion of 1 glucose to 3 acetylphosphates and the alternative implementations of the well-known autocatalytic Formose process.

The approach presented here combines aspects of previous pathway models. It can be used for finding single pathways as well as for the targeted, large-scale enumeration of pathways, both with respect to a given optimisation criterion, but without the burden of characterising the complete solution space.

2 Model

In this section we develop a formal model for pathways in reaction networks. For conciseness, we entirely use the language of hypergraphs. We first describe the basic pathway model and a simple notion of autocatalysis, characterised by the signature of the overall reaction of a pathway. This model allows for misleading, or chemically "uninteresting", pathways. An expanded model is therefore introduced to narrow the space of pathways.

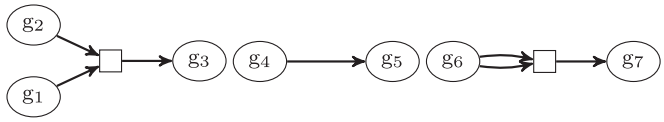

Fig. 1. Visualization of an abstract reaction network consisting of the reactions $\mathrm{g}_1 + \mathrm{g}_2 \longrightarrow \mathrm{g}_3$, $\mathrm{g}_4 \longrightarrow \mathrm{g}_5$, and $2\,\mathrm{g}_6 \longrightarrow \mathrm{g}_7$. Note that we use a simplified visualization scheme for 1-to-1 reactions, without a box but simply with a direct arrow between the compounds.

Note that the core model aims at formalising the notion of a chemical pathway, and therefore does not by itself include any optimality criteria. This model forms the basis of a practical implementation in terms of integer linear programs, which we will describe in detail in the following section. Different optimality criteria can then be added to the model in the form of linear objective functions.

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 $e^- \subseteq V$, called the tail and head of e. In the following we refer to directed multi-hypergraphs simply as hypergraphs. Fig. 1 gives an example of the visualisation scheme we generally use throughout this contribution.

Since we will use both normal sets and multisets we introduce the following notation for multisets. When constructing a multiset we use double curly brackets (i.e., $\{\!\{\ldots\}\!\}$) to distinguish them from normal set constructors, $\{\ldots\}$. 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., $\sum_{q \in_m Q} 1 = m_q(Q) = |\{\!\{q \in_m Q\}\!\}|$.

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

We need a mechanism to inject and extract molecules from the network, i.e., what is called "exchange reactions" in metabolic networks. We therefore define the *extended hypergraph* $\overline{\mathcal{H}} = (V, \overline{E})$ of $\mathcal{H}$ with $\overline{E} = E \cup E^- \cup E^+$ where $E^- = \{e_v^- = (\emptyset, \{\!\{v\}\!\}) \,|\, v \in V\}$ and $E^+ = \{e_v^+ = (\{\!\{v\}\!\}, \emptyset) \,|\, v \in V\}$ designate the additional "half-edges" e_v^- and e_v^+, for each $v \in V$. An example is shown in Fig. 2.

For a vertex $v \in V$ and an edge $e \in \overline{E}$ we use the multiplicity function for multisets to 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., $\delta^+_{\overline{E}}(v) = \{e \in \overline{E} \,|\, v \in e^+\}$. Likewise, $\delta^-_{\overline{E}}(v)$ denotes the restricted set of incident in-edges of v.

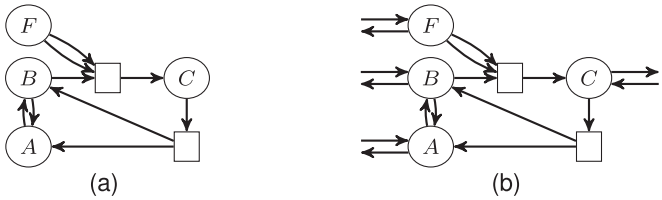

Fig. 2. (a) A small network, $\mathcal{H}$. (b) The extended network, $\overline{\mathcal{H}}$. Note that most of the input/output edges in the extended network will be constrained in the final formulation, and thus for many chemical networks many of these edges will effectively be removed to model specific interface conditions.

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

$$\sum_{e \in \delta^+_{\overline{E}}(v)} m_v(e^+) f(e) - \sum_{e \in \delta^-_{\overline{E}}(v)} m_v(e^-) f(e) = 0. \tag{1}$$

That is, the sum of flow out of each vertex must be the same as the sum of flow into it. 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 a set of inputs (sources) $S \subseteq V$ and outputs (targets/sinks) $T \subseteq V$. Thus

$$f(e_v^-) = 0 \quad \forall v \notin S \quad \text{and} \quad f(e_v^+) = 0 \quad \forall v \notin T, \tag{2}$$

serve as additional constraints in an I/O-constrained extended hypergraph, which is completely specified by the 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

$$\sum_{i=1}^{|V|} f(e_{v_i}^-)\, v_i \longrightarrow \sum_{i=1}^{|V|} f(e_{v_i}^+)\, v_i.$$

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

Flows are non-negative by definition. 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.

Finally, as in many typical flow problems, a capacity function $u : \overline{E} \to \mathbb{N}_0$ can be added to limit the flow from above, i.e., $f(e) \leq u(e)$. We will, however, not explicitly make use of a 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 [30]). Usually several reactions are organised in a cyclic network to achieve the proper topology for autocatalysis.

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

$$0 < f(e_v^-) = f(e_v^+). \tag{3}$$

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

$$0 < f(e_v^-) < f(e_v^+). \tag{4}$$

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

2.4 Chemically Simple Flow and Vertex Expansion

Representing reversible reactions as pairs of irreversible ones gives rise to trivial pathways consisting of edge e and its inverse e^{-1} with equal positive flow. Consider the example in Fig. 3a. 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 [8], [10] to prune such cases. However, this approach seems quite difficult to control in practice. Disallowing all cases of flow on both a reaction

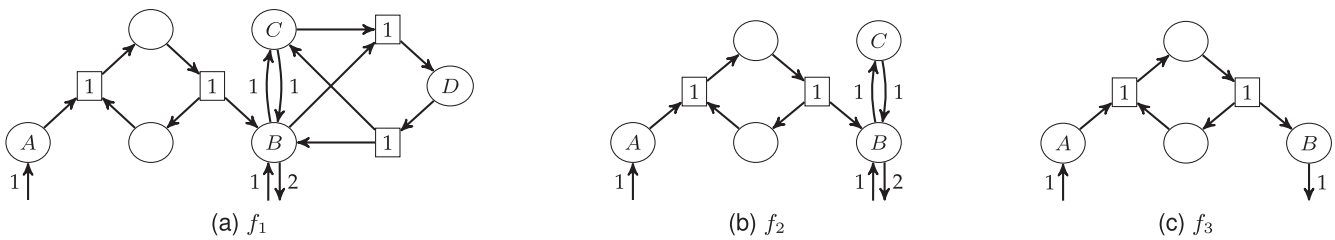

Fig. 3. Simplification of a flow f_1 to an equivalent flow f_3, by removal of futile 2-edge sub-pathways. (a) The molecule D is created through $B + C \longrightarrow D$, but can only be interpreted as being consumed through the reverse reaction. (b) After removal of 1 flow from the reactions $B + C \longleftrightarrow D$, 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.

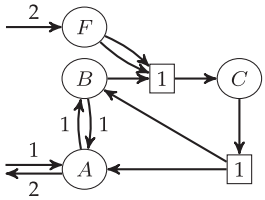

Fig. 4. Example of a flow with meaningful flow on an edge and its inverse, in the network from Fig. 2b. Only edges with non-zero flow are shown.

and its reverse as in [8], for example, turns out to be too restrictive. Another approach is to simply cancel out flow as a post-processing step [31]. In the following we illustrate that certain seemingly futile flows should be allowed. To facilitate precise constraints we therefore 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. 3b. The same reasoning can now be applied to C, and subsequently B, resulting in the flow depicted in Fig. 3c.

The original flow in Fig. 3a fulfils the requirement for overall autocatalysis in vertex B (Eq. (4)), but clearly the inflow of 1 B is not involved in the extra production of B, which goes against the idea of autocatalysis. The simplified flow, Fig. 3c, is however not overall autocatalytic, and it is therefore desirable to constrain the model such that futile 2-edge flows are not possible, in order to further approach a precise characterisation of autocatalysis.

From the shown example it is tempting to simply disallow flow on both an edge and its inverse. This is the approach effectively used in FBA-related methods, and also in flows on normal graphs [14], [32]. Since flow on an edge and its inverse is inherently a part of the I/O specification for autocatalysis, some exceptions must be made for I/O flow. However, as illustrated with the counterexample in Fig. 4 even internal edges can have seemingly futile 2-cycles.

We can interpret this flow such that no flow is directly reversed:

1) $\emptyset \longrightarrow A$
2) $A \longrightarrow B$
3) $\emptyset \longrightarrow F$ twice
4) $B + 2\,F \longrightarrow C$
5) $C \longrightarrow A + B$
6) $B \longrightarrow A$
7) $A \longrightarrow \emptyset$ twice

Since this interpretation has the flow on mutually inverse edges temporally separated, it should not be excluded from the pathway model.

To facilitate fine-grained control of flow where directly reversed flow can be disallowed 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 $v \in V$

$$V_v^- = \{u_{v,e}^- \,|\, \forall e \in \delta_{\overline{E}}^-(v)\} \tag{5}$$

$$\begin{aligned} V_v^+ &= \{u_{v,e}^+ \,|\, \forall e \in \delta_{\overline{E}}^+(v)\} \\ E_v &= \left\{(\{\!\{u^-\}\!\}, \{\!\{u^+\}\!\}) \,|\, u^- \in V_v^-, u^+ \in V_v^+\right\}. \end{aligned} \tag{6}$$

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 $e = (e^+, e^-) \in \overline{E}$ the reconnected edge is $\widetilde{e} = (\widetilde{e}^+, \widetilde{e}^-)$ with $\widetilde{e}^- = \{\!\{u_{v,e}^- \,|\, v \in_m e^-\}\!\}$ and $\widetilde{e}^+ = \{\!\{u_{v,e}^+ \,|\, v \in_m e^+\}\!\}$. We finally define the expanded hypergraph $\widetilde{\mathcal{H}} = (\widetilde{V}, \widetilde{E})$ as

$$\widetilde{V} = \bigcup\nolimits_{v \in V} V_v^- \cup \bigcup\nolimits_{v \in V} V_v^+ \quad \text{and} \quad \widetilde{E} = \bigcup\nolimits_{v \in V} E_v \cup \{\widetilde{e} \,|\, e \in \overline{E}\}.$$

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

$$\sum_{e \in \delta_{\widetilde{E}}^+(v)} m_v(e^+) f(e) - \sum_{e \in \delta_{\widetilde{E}}^-(v)} m_v(e^-) f(e) = 0. \tag{7}$$

The I/O constraints translates directly to the expanded network. In Section 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: for each pair of mutually inverse edges $e = (e^+, e^-), e' = (e^-, e^+) \in \overline{E}$ we consider the transit edges they induce, i.e., the edges $(u_{v,e}^-, u_{v,e'}^+)$ for all $v \in e^-$. None of these edges may have flow through them. A flow f must thus satisfy

$$f((u_{v,e}^-, u_{v,e'}^+)) = 0 \qquad \forall v \in e^-. \tag{8}$$

Fig. 5 shows the expanded version of the network from Fig. 2b, 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. 4 shows a valid flow with a meaningful 2-cycle. The expanded flow in Fig. 5 no longer has 2-cycles.

In Eqs. (3) and (4) we defined the I/O constraints for overall catalysis and autocatalysis. These constraints are converted in the obvious manner to the expanded network $\widetilde{\mathcal{H}}$. However, the expanded network reveals another possibility for somewhat misleading flows, exemplified in Fig. 6a. 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. 6b.

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 (3) (resp. overall autocatalysis (4)). In the expanded network f must additionally satisfy the transit constraints (note that $\delta_E^+(v)$ does not include the I/O edges)

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

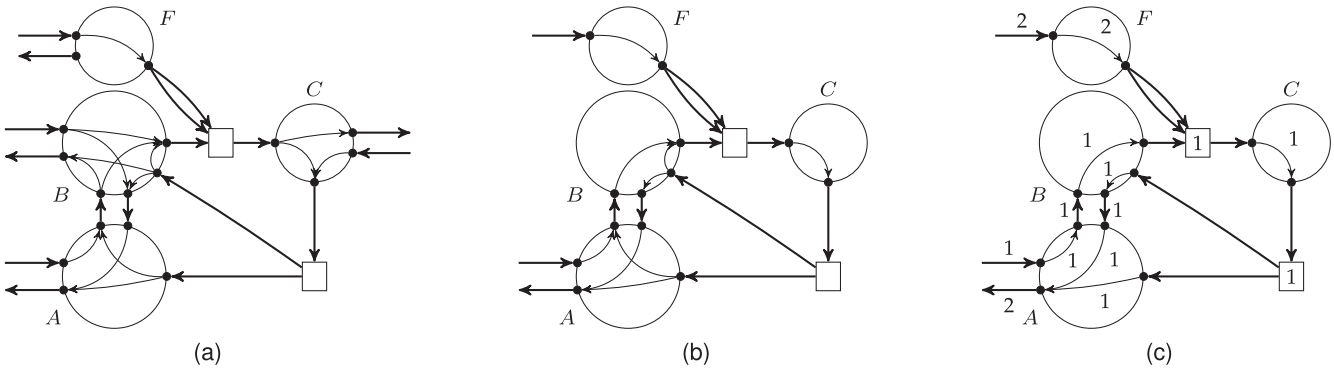

Fig. 5. The network from Fig. 2b expanded into $\widetilde{\mathcal{H}}$. The vertices of $\widetilde{\mathcal{H}}$ are the small filled circles, while the large circles, A, B, C, and F, only serves as visual grouping of the actual vertices. (a) The expanded network without transit edges constrained in Eq. (8). (b) The expanded network, simplified using the source set $S = \{A, F\}$ and sink set $T = \{A\}$. (c) The example flow from Fig. 4 in the expanded network, with only edges with non-zero flow shown. Note that no 2-cycles exist in this flow.

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^+.

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. 7. As a variant of our definition of overall 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, $S \backslash \{v\}$. 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.

Algorithm 1. Breadth-First Marking of a Hypergraph

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

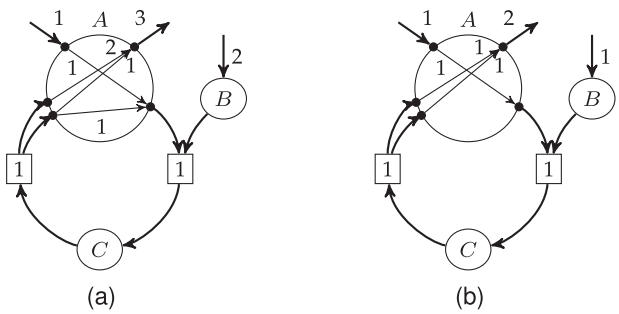

Fig. 6. A simplified network with an overall autocatalytic flow. (a) The vertex A is both overall autocatalytic and an intermediate vertex in the flow. (b) The same motif for overall autocatalysis, but without A being an intermediate vertex.

Note that breadth-first marking of hypergraphs, and variations thereof, has in the literature also been referred to as finding *scopes* of molecules [33]. 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 [34]. The methods thus do 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. In this section we only state the results. The proofs can be found in the supplemental material, which can be found on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TCBB.2017.2781724.

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 represented by a bipartite normal graph, then this size corresponds to the number of vertices and edges.

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

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

Proposition. *Let $f : \overline{E} \to \mathbb{N}_0$ 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 $g : \widetilde{E} \to \mathbb{N}_0$ in $\widetilde{\mathcal{H}}$ exists such that $g(\widetilde{e}) = f(e)$ for all $e \in \widetilde{E}$. If it exists it can be computed in polynomial time.*

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 [18]. The last

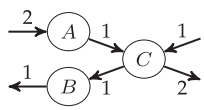

Fig. 7. The vertex C is overall autocatalytic, but not autocatalytic in the chemical sense.

proposition underlines that the expansion of the network does not drastically increase the complexity of finding pathways, but the proof of it also shows a potentially practical algorithmic approach to working with flows in the expanded network. In Section 2.7 we will however show an approach to directly find the flows in the expanded network using ILP.

2.6 Comparison to Existing Methods

The basic pathway model described in Section 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 [35]. The basic model is also similar to the model presented in [8], although the specification of allowed I/O flow is phrased differently. We briefly recast FBA in terms of hypergraphs as the underlying model 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 Section 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, $V = \{v_1, \ldots, v_{|V|}\}$ and $E = \{e_1, \ldots, e_{|E|}\}$. 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 $\overline{\mathcal{H}}$.

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 [36], [37] currently intensively investigates multi-enzyme cascade reactions, i.e., the combination of several enzyme reactions in concurrent one-pot processes [38], 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.

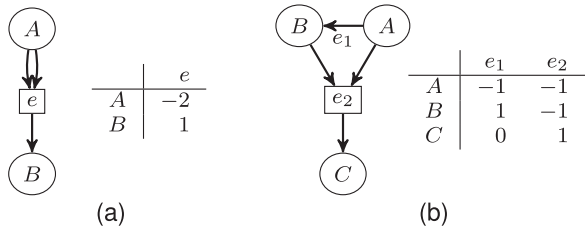

Fig. 8. Examples of reaction networks with not totally unimodular stoichiometric matrices. (a) All entries in a TU matrix must be -1, 0, or $+1$. (b) The submatrix consisting of the top two rows has determinant 2.

A *flux vector* $f \in \mathbb{R}^{|E|}$ models a pathway, and must satisfy the usual conservation constraint, $\mathbf{S} \cdot f = 0$ (cmp. Eq. (1)). Reversible reactions are modelled in one of two ways:

(i) 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 non-negative. This is the approach followed when finding EFMs [5].

(ii) Separate: reversible reactions are modelled as two inverse reactions, and the flow/flux on all reactions must be non-negative. This is the approached followed when finding ExPas [39]. 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. (2), specified by S and T, translate naturally to the corresponding constraints on the extended flux vector.

With FBA we additionally define a linear objective function to find an optimal flux vector, possibly with additional linear constraints [40]. As a flux vector is real-valued, and all the stated constraints are linear, it can be found using linear programming in polynomial time [15], [16]. Herein lies a major difference to the model presented in this contribution, 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 yields 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 [14], [32]. As the simple examples in Fig. 8 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 Section 3.1, 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

TABLE 1
A Non-Integer Pathway with Maximum Production of AcP from 1 X5P

Flow	Reaction
1.0	$\mathrm{G3P} + \mathrm{DHAP} \longrightarrow \mathrm{FBP}$
1.0	$\mathrm{G3P} \longrightarrow \mathrm{DHAP}$
0.5	$\mathrm{R5P} \longrightarrow \mathrm{X5P}$
0.5	$\mathrm{E4P} + \mathrm{F6P} \longrightarrow \mathrm{G3P} + \mathrm{S7P}$
1.5	$\mathrm{P_i} + \mathrm{X5P} \longrightarrow \mathrm{AcP} + \mathrm{G3P} + \mathrm{H_2O}$
0.5	$\mathrm{P_i} + \mathrm{F6P} \longrightarrow \mathrm{AcP} + \mathrm{E4P} + \mathrm{H_2O}$
0.5	$\mathrm{P_i} + \mathrm{S7P} \longrightarrow \mathrm{AcP} + \mathrm{R5P} + \mathrm{H_2O}$
1.0	$\mathrm{FBP} + \mathrm{H_2O} \longrightarrow \mathrm{P_i} + \mathrm{F6P}$
Overall	$\mathrm{X5P} + 1.5\ \mathrm{P_i} \longrightarrow 2.5\ \mathrm{AcP} + 1.5\ \mathrm{H_2O}$

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 the supplemental material, available online, for a table of molecule abbreviations.*

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 produces just 1 AcP, by the single reaction $\mathrm{P_i} + \mathrm{X5P} \longrightarrow \mathrm{AcP} + \mathrm{G3P} + \mathrm{H_2O}$. However, the optimal solution to the LP relaxation of the problem yields 2.5 AcP, via the pathway described in Table 1. This non-integer pathway is incidentally a superposition of the optimal integer pathway and a recycling pathway for converting 1 G3P into 1.5 AcP.

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 possibly 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, as low as 100 copies [41], and even smaller for biological macromolecules [42], [43]. Small numbers may have a profound influence on chemical kinetics and can in many cases cause qualitative changes in the behaviour of systems [44], [45]. 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. On the other hand, in some cases where composite pseudo-reactions are used to model experimental results, as in the case of biomass as objective function, non-integer coefficients may appear in way that makes scaling difficult or undesirable. Such models, however, inherently are are approximations consistent with the LP relaxation. Mechanistic pathways are no longer well-defined in such as setting.

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, available online, 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

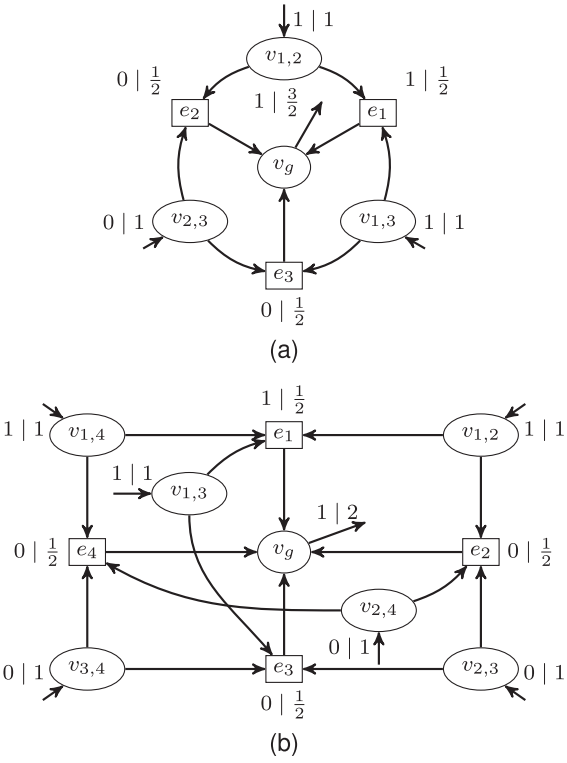

Fig. 9. Reduction from the independent set problem to the problem of maximizing output from a molecule in a reaction network, applied to the two graphs (a) K_3 and (b) K_4. The hyperedges are annotated with both a feasible integer flow and a feasible non-integer flow, written as $\langle \text{integer} \rangle \,|\, \langle \text{non-integer} \rangle$. Allowing a maximum inflow of 1 to all vertices $v_{i,j}$ and maximizing the outflow of v_g corresponds to finding a maximum independent set in the original graph.

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. 9. This illustrates that the use of the LP relaxation is not just a technical detail, but changes the nature of the problem entirely.

2.7 Implementation Using Integer Linear Programming

The ILP formulation characterising feasible flows is based on an expanded hypergraph $\widetilde{\mathcal{H}} = (\widetilde{V}, \widetilde{E})$. The flow function is modelled by an integer variable x_e for each edge $e \in \widetilde{E}$, and by constraints for flow conservation. The basic constraints are thus $x_e \in \mathbb{N}_0$ for all $e \in \widetilde{E}$ and

$$\sum_{e \in \delta^+_{\widetilde{E}}(v)} m_v(e^+) \cdot x_e - \sum_{e \in \delta^-_{\widetilde{E}}(v)} m_v(e^-) \cdot x_e = 0.$$

This definition is similar to an ILP formulation of a classical network flow problem, but with important differences; $\widetilde{H}$ is a hypergraph so an edge $e \in \widetilde{E}$ may be in both $\delta^-_{\widetilde{E}}(u)$ and $\delta^-_{\widetilde{E}}(v)$ (or $\delta^+_{\widetilde{E}}(u)$ and $\delta^+_{\widetilde{E}}(v)$) for $u \neq v$. Additionally, $\widetilde{H}$ 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. (8) are added in the obvious way. However, to reduce the size of the ILP we merge transit nodes where no incident transit edges have associated constraints, i.e., those created from irreversible reactions.

In the following sections we describe constraints for finding overall catalytic and autocatalytic flows. For the formulation we will use M to denote a classical "large enough" constant, and as some parts of the models for overall catalysis and autocatalysis are similar we will first describe the formulation of these common parts.

In many cases of practical relevance not only the optimal solution is relevant because alternative, near optimal solutions may be easier to find and realize in an evolutionary context. 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 [46]. This has also been applied to FBA [47]. The restriction of ILP keeps the set of candidate solution countable (even finite with flow capacity constraints), and it becomes straight-forward to enumerate not only optimal solutions, but also near-optimal solutions, see Section 2.8.

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 constraints $x_e \leq M \cdot (1 - z_v)$ for all $e = (u^-_{v,e'}, u^+_{v,e''}) \in V^-_v \times V^+_v : e' \neq e^-_v \wedge e'' \neq e^+_v$.

We model catalysis by introducing an indicator variable $z^c_v \in \{0, 1\}$ 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 $\sum_{v \in V} z^c_v \geq 1$. The actual constraints for the indicator variables are obtained partially by the constraints above on strictness of flow. Below follows the last requirement, Eq. (3), which is realised through a set of auxiliary indicator variables, $z^0_v, z^<_v, z^>_v \in \{0, 1\}$

$$x^-_v = x^+_v = 0 \Leftrightarrow z^0_v = 1 \equiv \begin{cases} 1 - z^0_v \leq x^-_v + x^+_v \\ M \cdot (1 - z^0_v) \geq x^-_v + x^+_v \end{cases}$$

$$x^-_v < x^+_v \Leftrightarrow z^<_v = 1 \equiv \begin{cases} x^-_v < x^+_v + M \cdot (1 - z^<_v) \\ x^-_v \geq x^+_v - M \cdot x^<_v \end{cases}$$

$$x^+_v < x^-_v \Leftrightarrow z^>_v = 1 \equiv \begin{cases} x^+_v < x^-_v + M \cdot (1 - z^>_v) \\ x^+_v \geq x^-_v - M \cdot z^>_v \end{cases}$$

$$0 < x^-_v = x^+_v \Leftrightarrow z^c_v = 1 \equiv \begin{cases} z^c_v \geq 1 - z^<_v - z^>_v - z^0_v \\ z^c_v \leq 1 - z^0_v \\ z^c_v \leq 1 - z^<_v \\ z^c_v \leq 1 - z^>_v . \end{cases}$$

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

We use the constraints for strictness of flow and model the remaining constraint, Eq. (4), using the auxiliary variable set z^-_v, indicating $x^-_v > 0$

$$0 < x^-_v \Leftrightarrow z^-_v = 1 \equiv \begin{cases} z^-_v \leq x^-_v \\ M \cdot z^-_v \geq x^-_v \end{cases}$$

$$0 < x^-_v < x^+_v \Leftrightarrow z^a_v = 1 \equiv \begin{cases} z^a_v \leq x^-_v \\ x^-_v < x^+_v + M \cdot (1 - z^a_v) \\ M \cdot z^a_v + x^-_v \geq x^+_v - M \cdot (1 - z^-_v). \end{cases}$$

2.8 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.

2.9 Software

The pathway model is implemented as an extension of the larger software package, MedØlDatschgerl (MØD) [48], [49], which can be downloaded at http://mod.imada.sdu.dk/. The software combines the pathway analysis with methods for working with generative chemistries [50], [51], including the algorithms for network expansion [52] 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 [53], or hypothetical prebiotic chemistries [54], [55]. 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, and a preliminary version is accessible at http://mod.imada.sdu.dk/playground.html, with examples of usage. The integrality constraints and the implicit constraints depending on this assumption can easily be disabled in our implementation. In this mode the framework is no longer guaranteed to produce mechanistic pathways. Instead, it reproduces classical FBA. Solution enumeration, as described above, is not well-defined in this setting.

3 Application Scenarios

Here we illustrate the strength of our modelling framework by analysing two specific chemical systems. We shall see

that beyond well studied objectives such as the minimization of the number of reactions used or maximizing the yield of a target compound, the objective can also be augmented with constraints for a chemical transformation pattern, for example overall autocatalysis. Both chemistries are modelled using the graph grammar approach described in [49], [50], [51], [52]. A visualisation of all rules can be found in the supplemental material, available online, along with tables of molecule name abbreviations.

3.1 Carbon Rearrangement in the Non-Oxidative Glycolysis Pathway

Microbial synthesis of plant-derived secondary metabolites currently is a hot topic in metabolic engineering [56], [57]. Impressive examples include the synthesis of the anti-malarial drug precursor artemisinic acid [58], and of (S)-reticuline [59], an important intermediate towards the benzylisoquinoline alkaloids codein or morphine. The idea of engineering microbes to synthesize useful compounds also extends to the production of fuel. One approach is to couple the catabolic pathway known as glycolysis, which decomposes glucose (a C_6 molecule) into acetyl coenzyme A units (AcCoA, a C_2 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 CO_2 during normal oxidative glycolysis, which pushes the yield of this pathway down to 75 percent, in terms of atom economy. A lossy process for producing educts for the synthetic biofuel producing pathway is naturally a bad idea from an economic perspective.

A recent study [60] 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 C_2 body and a C_4 body as putative wast, to a carbon rearrangement network, which then recycles the C_4 body into molecules, that can be fed back into the NOG as educts. With this strategy NOG achieves a 100 percent 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 [60] 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 the 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 C_5 sugar) or sedoheptulose 1,7-bisphosphate (SBP, a C_7 sugar). Three pathways are shown (Fig. 2, [60]) that exploits this freedom in the design of the NOG pathway motif. In this section we illustrate that many more equivalent solutions can be found automatically, using enumeration of mechanistic pathways. In order to explore related reactions for which concrete enzymes may not yet exist, we use a generic model of the chemistry.

The molecules are encoded as graphs in the straightforward manner, though without stereochemical information.

TABLE 2
List of Generic Transformation Rules for Modelling the Non-Oxidative Glycolysis Chemistry

Abbr.	Name	Description
AL	Aldolase	A generic aldol addition.
AlKe	Aldose-Ketose	Aldehyde to ketone conversion.
KeAl	Ketose-Aldose	Ketone to aldehyde conversion.
PHL	Phosphohydrolase	Use water to cleave off phosphate.
PK	Phosphoketolase	Break C-C-bond and add phosphate.
TAL	Transaldolase	Move C_3-end.
TKL	Transketolase	Move C_2-end.

The full details of the rules are shown in the supplemental material, available online.

This implies that certain classes of molecules are represented as a single molecule, e.g., $\mathrm{Ru5P}$ and $\mathrm{X5P}$.

We have modelled the generic transformation rules listed in Table 2, and shown in the supplemental material, available online. In [60] the use of phosphoketolase is associated with specific names for the specific reactions:

$$\text{XPK for } \mathrm{X5P} + \mathrm{P_i} \longrightarrow \mathrm{AcP} + \mathrm{G3P}$$
$$\text{FPK for } \mathrm{F6P} + \mathrm{P_i} \longrightarrow \mathrm{AcP} + \mathrm{E4P}$$

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

$$\text{SPK for } \mathrm{S7P} + \mathrm{P_i} \longrightarrow \mathrm{AcP} + \mathrm{X5P}$$
$$\text{OPK for } \mathrm{C_8P} + \mathrm{P_i} \longrightarrow \mathrm{AcP} + \mathrm{G6P}$$

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

The network is created by iterating the application of all the transformation rules listed in Table 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 C_8 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 [61]). The network generation therefore terminates, and results in a network with 81 molecules and 414 reactions. The ILP for finding pathways contains 12,077 variables, of which 10,477 are due to transit edges. Without transit node merging the number of transit edges would have been 23,805.

For the overall reaction $\mathrm{F6P} + 2\,\mathrm{P_i} \longrightarrow 3\,\mathrm{AcP} + 2\,\mathrm{H_2O}$ 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 [62]. This results in 263 different pathways, which were computed in 5 minutes on a desktop computer with an Intel Core i7-6700 CPU (3.40 GHz) . In Table 3 we categorise the pathways w.r.t. (i) the number of unique reactions used, (ii) the number of reactions, (iii) whether the only

TABLE 3
Overview of Number of NOG Pathways

	Only FBP				Other Bisphosphates							
	8 Unique React.				7 Unique React.				8 Unique React.			
PK Type	Reactions				Reactions				Reactions			
X, F, S, O	8	9	10	11	7	8	9	10	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

Categories marked with subscript $_a$, $_b$, and $_c$ refer to Fig. 2 of [60], 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 Table 4. The pathway shown in Fig. 10a is from the framed blue category, and the pathway in Fig. 10b is from the framed green category. Their counterparts with the replacement from Table 4 are the unframed blue and green numbers.

TABLE 4
Two Pathways with the Same Overall Reaction

$$\begin{aligned} \mathrm{FBP} + \mathrm{H_2O} &\xrightarrow{\mathrm{PHL}} \mathrm{P_i} + \mathrm{F6P} \\ \mathrm{E4P} + \mathrm{F6P} &\xrightarrow{\mathrm{TAL}} \mathrm{G3P} + \mathrm{S7P} \\ \mathrm{G3P} + \mathrm{DHAP} &\xrightarrow{\mathrm{AL}} \mathrm{FBP} \\ \hline \mathrm{DHAP} + \mathrm{E4P} + \mathrm{H_2O} &\longrightarrow \mathrm{S7P} + \mathrm{P_i} \end{aligned}$$

(a)

$$\begin{aligned} \mathrm{DHAP} + \mathrm{E4P} &\xrightarrow{\mathrm{AL}} \mathrm{C_7P_2} \\ \mathrm{H_2O} + \mathrm{SBP} &\xrightarrow{\mathrm{PHL}} \mathrm{S7P} + \mathrm{P_i} \\ \hline \mathrm{DHAP} + \mathrm{E4P} + \mathrm{H_2O} &\longrightarrow \mathrm{S7P} + \mathrm{P_i} \end{aligned}$$

(b)

(a) A 3-reaction pathway using FBP as bisphosphate. This sub-pathway is highlighted in Fig. 10a. (b) a 2-reaction pathway using a bisphosphate with 7 carbons. This sub-pathway is highlighted in Fig. 10b.

bisphosphate used is FBP, and (iv) the histogram of different PK reactions (see the modelling above). The table shows the number of solutions for 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 Table 4.

In Fig. 10a one of the solutions is illustrated in detail. This solution has similar properties to the solution shown in Fig. 2a in [60]: 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. 10b 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.

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.

3.2 Overall Autocatalysis in the Formose Process

Carbohydrates $(\mathrm{CH_2O})_n$, vulgo sugars, can formally be viewed as polymers of formaldehyde $\mathrm{CH_2O}$ building blocks. The chemical reactivity of sugars is dominated by the two functional groups (1) carbonyl group ($\mathrm{C=O}$) and (2) vicinal hydroxyl groups ($\mathrm{HO-C-C-OH}$), 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 [63] 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 [30]). The formose process has intensively been studied (for a recent review see [64]) for its potential to produce biologically significant carbohydrates from formaldehyde under prebiotic conditions [65]. 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 $\mathrm{C_4}$–$\mathrm{C_6}$ 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 [25], [64], [66], 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 [66], [67] that this base cycle cannot account for the massive consumption of formaldehyde, after the lag-phase. The reasons is that, under the 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

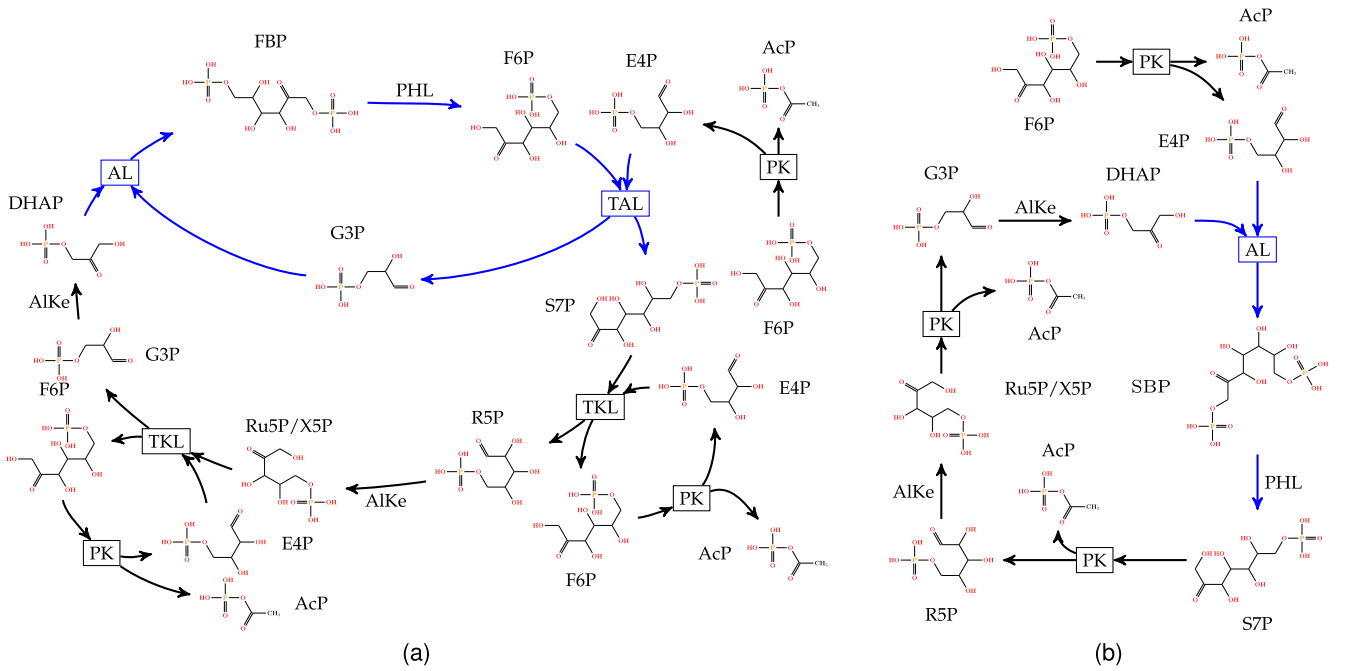

Fig. 10. Two solution pathways for NOG. (a) A pathway similar to [60, Fig. 2a], but using fewer reactions. This solution category is the framed blue cell of Table 3. The highlighted sub-pathway is the pathway from Table 4a. (b) The shortest pathway found, denoted by the framed green cell in Table 3. The highlighted sub-pathway is the pathway from Table 4b.

following mechanistic picture arises; fast repeated addition of formaldehyde to enolized keto groups produces larger sugars, draining material from the base cycle and retro-aldol fragmentation of larger sugars replenishes the base cycle (or variants) with short carbohydrates. It is unknown how these higher order cycles are structurally organized around the base autocatalytic cycle and how they are interconnected.

For the formose chemistry we adopt the following naming scheme for molecules; $\mathrm{C}_{N\langle t\rangle}$, where N specifies the number of carbon atoms and $\langle t\rangle$ indicates the position of the double bond. We use $_{\mathrm{a}}$ for aldehydes, $_{\mathrm{e}}$ for enol forms, and $_{\mathrm{k}}$ for ketones. The model follows [65] using only two basic types of reactions, keto-enol tautomerism and aldol reaction. Both are reversible, thus giving the four transformation rules listed in Table 5 and the supplemental material, available online. Starting molecules are formaldehyde (C_1) and glycolaldehyde ($\mathrm{C}_{2\mathrm{a}}$). 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 was 8 seconds on a Intel Core i7-6700 CPU (3.40 GHz) . The ILP for finding pathways contains 33,241 variables, of which 28,240 are due to transit edges. In this network all reactions are reversible and no transit node merging can thus been performed.

TABLE 5
List of Generic Transformation Rules for Modelling the Formose Chemistry

Abbr.	Name	Description
KeEn	Keto-Enol	Keto to enol form conversion.
EnKe	Enol-Keto	Enol to keto form conversion.
AA	Aldol addition	Merge an enol and a keto.
RAA	Retro-aldol add.	Split a keto into enol and keto.

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

$$\mathrm{C}_{2\mathrm{a}} + 2\ \mathrm{C}_1 \longrightarrow 2\ \mathrm{C}_{2\mathrm{a}},$$

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. Table 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, using a computing nodes with Intel Xeon E5-2665 CPUs (2.4 GHz). The enumeration procedure issued in total 1,565,756 optimisation queries to CPLEX, arising from a search trees of a combined size of

TABLE 6
Overview of the Number of Autocatalytic Flows in the Formose Chemistry

	Maximum #C						
Unique reactions used	4	5	6	7	8	9	Sum
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

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). Four of the smallest pathways can be found in the supplemental material, available online.

3,129,218 nodes. The unknown entries in the table are due to high memory demands (more than 64 GB) for single enumeration runs. A selection of pathways can be found in the supplemental material, available online.

The computational analysis of the chemical space of the formose process 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 overall 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 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 C_2/C_3 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.

4 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 ILP. This approach also enables a targeted enumeration of pathways of interest. Naturally, such systematic enumerations can become computationally much more challenging than finding a single optimal solution. However, on the other hand, it allows for exploring the space of inferred mechanisms.

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.

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 workflow rests on directed evolution and screening [68]. 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 [56], [58], [69], this strategy must fail, if no natural pathway is known that implements the desired transformation.

The pathway modelling framework presented here is implemented as part of a larger software package that includes the methods for automatic generation of reaction networks using graph transformation [48], [49]. As illustrated with both the formose and NOG chemistries, this combination results in an extremely powerful framework for attacking a wide range of problems from Chemistry and Biology. The grounding of the graph transformation approach in well established mathematical theory allows us to provide efficient algorithms for an intermediary level of detail. For instance, it 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 [51], thereby enabling precise computer-assisted coarse graining operations on reaction networks.

In recent years the interest in CTMs resurfaced in the context of designing alternative networks performing the same function as their natural archetypes [70], 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 [71], recent work extends the original approaches to larger networks including diverse reaction chemistry such as the central carbon metabolism [72] or the CO_2 fixation pathways [73]. In this literature the concept of CTM is only discussed implicitly and to our knowledge there has been no explicit attempt to formalize CTMs.

Here, we have illustrated how higher-level CTMs can be detected efficiently in given reaction networks using overall (auto)catalytic pathways as an example. This is possible whenever input and output of a motif can be specified as a linear constraint, which includes in particular sets of allowed input molecules and desired output molecules. In combination with the graph transformation formalism our framework is even capable of designing alternative pathways enforcing, e.g., the use of prescribed intermediates.

Last but not least, the use of directed multi-hypergraphs is sufficient to strictly enforce chemical constraints. At the same time they make complex network transformations convenient to understand, since hyperflows correspond to subnetworks that have intuitively understandable graphical representations.

ACKNOWLEDGMENTS

This work was supported in part by the Volkswagen Stiftung proj. no. I/82719, the COST-Action CM1304 "Systems Chemistry", and by the Danish Council for Independent Research, Natural Sciences, grants DFF-1323-00247 and DFF-7014-00041. 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.

REFERENCES

[1] A. V. Zeigarnik, "On hypercycles and hypercircuits in hypergraphs," *Discrete Math. Chemistry*, vol. 51, pp. 377–383, 2000.
[2] D. A. Fell and J. R. Small, "Fat synthesis in adipose tissue. An examination of stoichiometric constraints," *Biochemical J.*, vol. 238, pp. 781–786, 1986.
[3] K. J. Kauffman, P. Prakash, and J. S. Edwards, "Advances in flux balance analysis," *Current Opinion Biotechnol.*, vol. 14, pp. 491–496, 2003.
[4] J. D. Orth, I. Thiele, and B. Ø. Palsson, "What is flux balance analysis?" *Nature Biotechnol.*, vol. 28, pp. 245–248, 2010.
[5] S. Schuster and C. Hilgetag, "On elementary flux modes in biochemical reaction systems at steady state," *J. Biol. Syst.*, vol. 2, no. 02, pp. 165–182, 1994.
[6] S. Klamt and J. Stelling, "Combinatorial complexity of pathway analysis in metabolic networks," *Mol. Biol. Rep.*, vol. 29, pp. 233–236, 2002.
[7] S. Klamt and J. Stelling, "Two approaches for metabolic pathway analysis?" *Trends Biotechnol.*, vol. 21, pp. 64–69, 2003.
[8] J. E. Beasley and F. J. Planes, "Recovering metabolic pathways via optimization," *Bioinf.*, vol. 23, no. 1, pp. 92–98, 2007.
[9] F. J. Planes and J. E. Beasley, "Path finding approaches and metabolic pathways," *Discrete Appl. Math.*, vol. 157, no. 10, pp. 2244–2256, 2009.
[10] F. J. Planes and J. E. Beasley, "A critical examination of stoichiometric and path-finding approaches to metabolic pathways," *Briefings Bioinf.*, vol. 9, no. 5, pp. 422–436, 2008.
[11] G. Gallo, G. Longo, S. Pallottino, and S. Nguyen, "Directed hypergraphs and applications," *Discrete Appl. Math.*, vol. 42, no. 2/3, pp. 177–201, 1993.
[12] R. Cambini, G. Gallo, and M. G. Scutellà, "Flows on hypergraphs," *Math. Program.*, vol. 78, pp. 195–217, 1997.
[13] G. Gallo, C. Gentile, D. Pretolani, and G. Rago, "Max Horn SAT and the minimum cut problem in directed hypergraphs," *Math. Program.*, vol. 80, pp. 213–237, 1998.
[14] R. K. Ahuja, T. L. Magnanti, and J. Orlin, *Network Flows: Theory, Algorithms, and Applications*. Englewood Cliffs, NJ, USA: Prentice Hall, 1993.
[15] L. G. Khachiyan, "Polynomial algorithms in linear programming," *USSR Comput. Math. Math. Physics*, vol. 20, no. 1, pp. 53–72, 1980.
[16] N. Karmarkar, "A new polynomial-time algorithm for linear programming," in *Proc. 16th Annu. ACM Symp. Theory Comput.*, 1984, pp. 302–311.
[17] R. M. Karp, "Reducibility among combinatorial problems," in *Complexity of Computer Computations*, R. E. Miller and J. W. Thatcher, Eds. New York, NY, USA: Plenum Press, 1972, pp. 85–103.
[18] J. L. Andersen, C. Flamm, D. Merkle, and P. F. Stadler, "Maximizing output and recognizing autocatalysis in chemical reaction networks is NP-complete," *J. Syst. Chemistry*, vol. 3, 2012, Art. no. 1.
[19] L. F. de Figueiredo, et al., "Computing the shortest elementary flux modes in genome-scale metabolic networks," *Bioinf.*, vol. 25, no. 23, pp. 3158–3165, 2009.
[20] S. Jonnalagadda and R. Srinivasan, "An efficient graph theory based method to identify every minimal reaction set in a metabolic network," *BMC Syst. Biol.*, vol. 8, no. 1, Mar. 2014, Art. no. 28.
[21] A. P. Burgard, S. Vaidyaraman, and C. D. Maranas, "Minimal reaction sets for escherichia coli metabolism under different growth requirements and uptake environments," *Biotechnol. Progress*, vol. 17, no. 5, pp. 791–797, 2001.
[22] A. Röhl and A. Bockmayr, "A mixed-integer linear programming approach to the reduction of genome-scale metabolic networks," *BMC Bioinf.*, vol. 18, no. 1, Jan. 2017, Art. no. 2.
[23] C. Kaleta, F. Centler, and P. Dittrich, "Analyzing molecular reaction networks: From pathways to chemical organizations," *Mol. Biotechnol.*, vol. 34, pp. 117–123, 2006.
[24] F. Centler, C. Kaleta, P. Speroni di Fenizio, and P. Dittrich, "Computing chemical organizations in biological networks," *Bioinf.*, vol. 24, pp. 1611–1618, 2008.
[25] K. Ruiz-Mirazo, C. Briones, and A. de la Escosura, "Prebiotic systems chemistry: New perspectives for the origins of life," *Chemical Rev.*, vol. 114, pp. 285–366, 2014.
[26] D. Blackmond, "An examination of the role of autocatalytic cycles in the chemistry of proposed primordial reactions," *J. Angew Chemie Int. Edition*, vol. 48, pp. 386–390, 2009.
[27] R. Plasson, A. Brandenburg, L. Jullien, and H. Bersini, "Autocatalyses," *J. Phys. Chemistry A*, vol. 115, no. 28, pp. 8073–8085, 2011.
[28] S. A. Kauffman, "Autocatalytic sets of proteins," *J. Theoretical Biol.*, vol. 119, no. 1, pp. 1–24, 1986.
[29] W. Hordijk, "Autocatalytic sets: From the origin of life to the economy," *BioSci.*, vol. 63, no. 11, pp. 877–881, 2013.
[30] A. J. Bissette and S. P. Fletcher, "Mechanisms of autocatalysis," *J. Angew Chemie Int. Edition*, vol. 52, no. 49, pp. 12 800–12 826, 2013.
[31] C. Kaleta, L. F. de Figueiredo, and S. Schuster, "Can the whole be less than the sum of its parts? Pathway analysis in genome-scale metabolic networks using elementary flux patterns," *Genome Res.*, vol. 19, no. 10, pp. 1872–1883, 2009.
[32] J. Bang-Jensen and G. Z. Gutin, "Digraphs: Theory, algorithms and applications," in *Springer Monographs in Mathematics*, Berlin, Germany: Springer, 2009.
[33] T. Handorf, O. Ebenhh, and R. Heinrich, "Expanding metabolic networks: Scopes of compounds, robustness, and evolution," *J. Mol. Evol.*, vol. 61, pp. 498–512, 2005, doi: 10.1007/s00239–005-0027-1.
[34] Á. Kun, B. Papp, and E. Szathmáry, "Computational identification of obligatorily autocatalytic replicators embedded in metabolic networks," *Genome Biol.*, vol. 9, 2008, Art. no. R51.
[35] J. A. Papin, J. Stelling, N. D. Price, S. Klamt, S. Schuster, and B. O. Palsson, "Comparison of network-based pathway analysis methods," *Trends Biotechnol.*, vol. 22, no. 8, pp. 400–405, 2004.
[36] E. García-Junceda, I. Lavandera, D. Rother, and J. H. Schrittwieser, "(Chemo)enzymatic cascades—Nature's synthetic strategy transferred to the laboratory," *J. Mol. Catalysis B: Enzymatic*, vol. 114, pp. 1–6, 2015.
[37] J.-M. Choi, S.-S. Han, and H.-S. Kim, "Industrial applications of enzyme biocatalysis: Current status and future aspects," *Biotechnol. Advances*, vol. 33, pp. 1443–1454, 2015.
[38] E. Ricca, B. Brucher, and J. H. Schrittwieser, "Multi-enzymatic cascade reactions: Overview and perspectives," *Adv. Synthesis Catalysis*, vol. 353, pp. 2239–2262, 2011.
[39] C. H. Schilling, D. Letscher, and B. Ø. Palsson, "Theory for the systemic definition of metabolic pathways and their use in interpreting metabolic function from a pathway-oriented perspective," *J. Theoretical Biol.*, vol. 203, no. 3, pp. 229–248, 2000.
[40] J. M. Savinell and B. Ø. Palsson, "Network analysis of intermediary metabolism using linear optimization. I. Development of mathematical formalism," *J. Theoretical Biol.*, vol. 154, pp. 421–454, 1992.
[41] P. Guptasarma, "Does replication-induced transcription regulate synthesis of the myriad low copy number proteins of escherichia coli?" *BioEssays*, vol. 17, no. 11, pp. 987–997, 1995.
[42] N. Fedoroff and W. Fontana, "Small numbers of big molecules," *Sci.*, vol. 297, pp. 1129–1130, 2002.
[43] R. Milo and R. Phillips, *Cell Biology by the Numbers*. New York, NY, USA: Garland Science, 2015.
[44] D. T. Gillespie, "Stochastic simulation of chemical kinetics," *Annu. Rev. Phys. Chemistry*, vol. 58, pp. 35–55, 2007.
[45] P. Kreyssig, C. Wozar, S. Peter, T. Veloz, B. Ibrahim, and P. Dittrich, "Effects of small particle numbers on long-term behaviour in discrete biochemical systems," *Bioinf.*, vol. 30, pp. i475–i481, 2014.
[46] G. Swart, "Finding the convex hull facet by facet," *J. Algorithms*, vol. 6, no. 1, pp. 17–48, 1985.
[47] S. Lee, C. Phalakornkule, M. M. Domach, and I. E. Grossmann, "Recursive MILP model for finding all the alternate optima in LP models for metabolic networks," *Comput. Chemical Eng.*, vol. 24, no. 2, pp. 711–716, 2000.

[48] J. L. Andersen, C. Flamm, D. Merkle, and P. F. Stadler, "A software package for chemically inspired graph transformation," in *Proc. 9th Int. Conf. Graph Transformation Memory Hartmut Ehrig*, 2016, pp. 73–88.
[49] J. L. Andersen, "MedØlDatschgerl (MØD)," 2017. [Online]. Available: http://mod.imada.sdu.dk
[50] J. L. Andersen, C. Flamm, D. Merkle, and P. F. Stadler, "Inferring chemical reaction patterns using graph grammar rule composition," *J. Syst. Chemistry*, vol. 4, 2013, Art. no. 4.
[51] J. L. Andersen, C. Flamm, D. Merkle, and P. F. Stadler, "50 shades of rule composition: From chemical reactions to higher levels of abstraction," in *Proc. 1st Int. Conf. Formal Methods Macro-Biol.*, 2014, pp. 117–135.
[52] J. L. Andersen, C. Flamm, D. Merkle, and P. F. Stadler, "Generic strategies for chemical space exploration," *Int. J. Comput. Biol. Drug Des.*, vol. 7, no. 2/3, pp. 225–258, 2014.
[53] J. L. Andersen, C. Flamm, M. M. Hanczyc, and D. Merkle, "Towards an optimal DNA-templated molecular assembler," in *Proc. 14th Conf. Synthesis Simul. Living Syst.*, 2014, pp. 557–564.
[54] J. L. Andersen, T. Andersen, C. Flamm, M. M. Hanczyc, D. Merkle, and P. F. Stadler, "Navigating the chemical space of HCN polymerization and hydrolysis: Guiding graph grammars by mass spectrometry data," *Entropy*, vol. 15, no. 10, pp. 4066–4083, 2013.
[55] J. L. Andersen, C. Flamm, D. Merkle, and P. F. Stadler, "In silico support for Eschenmoser's glyoxylate scenario," *Israel J. Chemistry*, vol. 55, no. 8, pp. 919–933, 2015.
[56] K. Thodey, S. Galanie, and C. D. Smolke, "A microbial biomanufacturing platform for natural and semisynthetic opioids," *Nature Chemical Biol.*, vol. 10, pp. 837–844, 2014.
[57] J. Nielsen, "Yeast cell factories on the horizon," *Sci.*, vol. 349, pp. 1050–1051, 2015.
[58] D.-K. Ro, et al., "Production of the antimalarial drug precursor artemisinic acid in engineered yeast," *Nature*, vol. 440, pp. 940–943, 2006.
[59] W. C. DeLoache, Z. N. Russ, L. Narcross, A. M. Gonzales, V. J. J. Martin, and J. E. Dueber, "An enzyme-coupled biosensor enables (s)-reticuline production in yeast from glucose," *Nature Chemical Biol.*, vol. 11, pp. 465–471, 2015.
[60] I. W. Bogorad, T.-S. Lin, and J. C. Liao, "Synthetic non-oxidative glycolysis enables complete carbon conservation," *Nature*, vol. 502, no. 7473, pp. 693–697, 2013.
[61] A. Stincone, et al., "The return of metabolism: Biochemistry and physiology of the pentose phosphate pathway," *Biol. Rev.*, vol. 90, pp. 927–963, 2015.
[62] R. J. Kleijn, W. A. van Winden, W. M. van Gulik, and J. J. Heijnen, "Reviciting the ^{13}C-label disribution of the non-oxidative branch of pentose phosphate pathway based upon kinetic and genetic evidence," *FEBS J.*, vol. 272, pp. 4970–4982, 2005.
[63] A. M. Butlerov, "Einiges über die chemische structur der Körper," *Zeitschrift für Chemie*, vol. 4, pp. 549–560, 1861.
[64] I. V. Delidovich, A. N. Simonov, O. P. Taran, and V. N. Parmon, "Catalytic formation of monosaccharides: From the formose reaction towards selective synthesis," *ChemSusChem*, vol. 7, no. 7, pp. 1833–1846, 2014.
[65] S. A. Benner, H.-J. Kim, M.-J. Kim, and A. Ricardo, "Planetary organic chemistry and the origins of biomolecules," *Cold Spring Harbor Perspectives Biol.*, vol. 2, no. 7, 2010, Art. no. a003467.
[66] A. Ricardo, F. Frye, M. A. Carrigan, J. D. Tipton, D. H. Powell, and S. A. Benner, "2-Hydroxymethylboronate as a reagent to detect carbohydrates: Application to the analysis of the formose reaction," *J. Organic Chemistry*, vol. 71, pp. 9503–9505, 2006.
[67] H.-J. Kim, et al., "Synthesis of carbohydrates in mineral-guided prebiotic cycles," *J. Amer. Chemical Soc.*, vol. 133, no. 24, pp. 9457–9468, 2011.
[68] P. M. Boyle and P. A. Silver, "Parts plus pipes: Synthetic biology approaches to metabolic engineering," *Metabolic Eng.*, vol. 14, pp. 223–232, 2012.
[69] E. Fossati, L. Narcross, A. Ekins, J.-P. Falgueyret, and V. J. J. Martin, "Synthesis of morphinan alkaloids in saccharomyces cerevisiae," *PLoS One*, vol. 10, no. 4, 2015, Art. no. e0124459.
[70] O. Ebenhöh and R. Heinrich, "Stoichiometric design of metabolic networks: Multifunctionality, clusters, optimization, weak and strong robustness," *Bulletin Math. Biol.*, vol. 65, pp. 323–357, 2003.
[71] E. Meléndez-Hevia, T. G. Waddell, and F. Montero, "Optimization of metabolism: The evolution of metabolic pathways towards simplicity through the game of the pentose phosphate cycle," *J. Theoretical Biol.*, vol. 166, pp. 201–220, 1994.
[72] E. Noor, E. Eden, R. Milo, and U. Alon, "Central carbon metabolism as a minimal biochemical walk between precursors for biomass and energy," *Mol. Cell*, vol. 39, pp. 809–820, 2010.
[73] A. Bar-Even, E. Noor, N. E. Lewis, and R. Milo, "Design and analysis of synthetic carbon fixation pathways," *Proc. Nat. Academy Sci. United States America*, vol. 107, pp. 8889–8894, 2010.

Jakob L. Andersen received the dual PhD degrees in computer science from the University of Southern Denmark and the University Leipzig, in 2015. He then joined the ELSI Origins Network (EON) as a research fellow in the Earth-Life Science Institute, Tokyo Institute of Technology, with an external postdoc affiliation in the Department of Mathematics and Computer Science, University of Southern Denmark.

Christoph Flamm received the doctorate degree in chemistry from the University of Vienna, in 1998. He was conferred the *venia docendi* in 2006 from the same school and is since then associate professor in the Institute for Theoretical Chemistry. His research focuses on questions at the boarder between chemistry and computer science.

Daniel Merkle received the diploma degree in computer science and the PhD degree in applied computer science from the University of Karlsruhe, Germany, in 1997 and 2002, respectively. He worked as an assistant professor in the Department of Computer Science, Leipzig, Germany, until April 2008. He was an associate professor in the Department of Mathematics and Computer Science, University of Southern Denmark until 2017 and has been promoted to a is full professor since then. His research interests include combinatorial and algorithmic approaches for chemistry.

Peter F. Stadler received the PhD degree in chemistry from the University of Vienna, in 1990 and then worked as an assistant and associate professor of theoretical chemistry at the same school. In 2002, he moved to Leipzig as a full professor of bioinformatics. Since 1994, he has been an external professor in the Santa Fe Institute. He has been an external scientific member of the Max Planck Society since 2009 and an corresponding member abroad of the Austrian Academy of Sciences since 2010.