Reaction Templates as a constraint in Program Synthesis of Chemical Reaction Networks

Alexandru-Daniel Dumitru

Supervisor(s): Sebastijan Dumančić, Reuben Gardos Reid

EEMCS, Delft University of Technology, The Netherlands

A Thesis Submitted to EEMCS Faculty Delft University of Technology,
In Partial Fulfilment of the Requirements
For the Bachelor of Computer Science and Engineering
June 20, 2026

Name of the student: Alexandru-Daniel Dumitru
Final project course: CSE3000 Research Project
Thesis committee: Sebastijan Dumančić, Reuben Gardos Reid, Jana Weber

Abstract

Recovery of full Chemical Reaction Networks from an incomplete (partially observable) problem is an extensive and combinatorial expensive process. This work investigates some of the possible methods to restrict the overall search space of program synthesis and obtain a better rank for the target network, through the use of reaction templates. As such, filter-based network pruning and ranking, as well as reaction size constraint are compared to the existing base synthesizer that this work builds upon, both in terms of search-space and target rank. Additionally, runtime is also considered as a second metric for these experiments. Overall, these methods have obtained some improvements in the small subset of benchmarked networks, but achieving integration of reaction templates into CRN program synthesis still remains an open challenge.

1 Introduction

Chemical Reaction Networks (CRNs) [1] represent one of the most fundamental observation and analysis tools available for scientists. They are mathematical models that describe the behavior of chemical reactions in a system, and are widely used in many domains such as pharmaceutical development, industrial chemistry, biology and material science. In these fields, CRNs can help with understanding certain reaction mechanisms, predict outcomes, or even improve yield and safety. The many explored and unexplored uses of CRNs have been highlighted in Pavel Loskot et al. [2], including biochemical systems used for analyzing cellular processes. Even more, their use in material science has been exemplified by applying similar models to CRNs in the understanding of catalytic reactions in Albert Bruixe et al.[3].

Typically, most of the research on CRNs has focused on the forward problem, where the goal is to predict the evolution of species' (reactants and products) concentrations given a known reaction network. However, in many real-world scenarios, reactions are not fully known, and further experiments or expert intervention are needed. This leads us to the inverse problem: given the evolution of species concentration over time and a partial chemical reaction network, recover all the missing reactions and species in order to reconstruct the whole network. Consequently, being able to solve this problem, not only more efficiently, but also in terms of obtaining more plausible networks, could help experts cut down the costs by having to run less chemical experiments.

An approach to this inverse problem is to use program synthesis [4], a subfield of AI that can automatically find a program given a language and some specification. Chemistry can be conveniently translated into specific grammars for molecules, reactions and networks. Thus, program synthesis can be used to solve this problem by enumerating molecules, then reactions and finally networks, given some initial molecules and concentrations. There has already been work done in this area in R. A. Wijers.' Master Thesis paper [5], in the form of synthesizing programs in a top-down search with constraints such as balanced reactions and ordered lists in order to reduce the search space. While this already shows promising results with the limitations imposed, scaling it to large or complex CRNs will still yield many networks with reactions that are not plausible.

Given how CRNs in the context of program synthesis are a relatively new idea, methods still lack strong chemical constraints and thus generate many implausible reactions. My proposed idea is to use program synthesis in these networks, but with more chemically inspired constraints. These can vastly reduce the search space of the synthesizer or rank the resulting networks by plausibility, as they can remove many reactions or molecules that do not make sense chemically.

One of the many constraints that can be used in this context are reaction templates [6], which encode common patterns of chemical reactions abstracted from known reactions. For example, a very known reaction is Water Formation: $2\,H_2 + O_2 \rightarrow 2\,H_2O$; which can be abstracted into the general addition template: $A + B \rightarrow AB$. Some of these patterns can be inferred directly into the synthesis procedure, either at the reaction or network level, or even be used to reconstruct possible molecules.

Therefore, in the following sections, we are going to look at how reaction templates can be implemented into a program synthesizer, both as a hard constraint (one that reduces search space and prunes candidate solutions that are not plausible) and as a soft constraint (sorting the candidate solutions by plausibility using heuristics). Furthermore, these methods will address the synthesizing process at the reaction level, as well as the network level. Then, based on each of these implementations, the following sub questions will be answered with experiments:

- Does the incorporation of reaction templates still allow recovery of previously known CRNs that are correct?
- Does the incorporation of reaction templates improve the rank of the expected CRN when compared to the base synthesizer?
- Does the incorporation of reaction templates constraint reduce the search space when compared to the base synthesizer?
- How does the runtime of the base synthesizer change with the implementation of reaction templates?

The Background and Related Work section provides an overview of the important topics and tools that are relevant to CRN reconstruction and reaction templates. In the Problem Definition section, the real-world problem of CRN recovery is stated formally. The Method section describes how reaction templates are implemented. The Experimental Setup and Results section states how each of the implementations are measured and the results, which are then reflected upon in the Discussion section. The Responsible Research section reflects upon ethical aspects, reproducibility and replicability. Lastly, the Conclusions and Future Work section reiterates the most important findings and suggests future work.

2 Background and Related Work

This chapter covers the main concepts that are relevant to this paper: Chemical Reaction Networks and Program Synthesis. Additionally, information about Reaction Templates are provided in order to help with the understanding of the Method section. Lastly, related work is addressed and put into contrast with the approaches in this paper.

2.1 CRNs

Chemical Reaction Networks are formally known as tools to describe how one or more reactions evolve over time in a system. These reactions are made up of species (reactants and products), for which the evolution of concentration over time is known. A high level visualization of this can be seen in Figure 1.

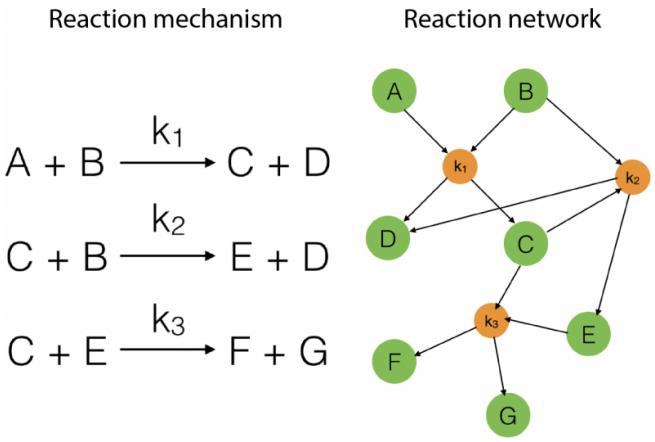

Figure 1: Example visualization of a Chemical Reaction Network, taken from this article. The capital letters (A-G) represent species and the k's are the rates at which the respective reactions occur.

A more detailed and in-depth description of CRN theory can be found in Martin Feinberg's work[7], including not only the basics of CRN theory, but also theorems and properties.

2.2 Program synthesis and code-base

Program synthesis represents the procedure of automatically generating programs that satisfy a certain grammar and rules, also known as specification. This paper builds on top of Richard Wijers' Thesis Paper and code-base [5], which is based on the Herb.jl program synthesis library written in Julia [8], but there are other approaches available as well: Stitch[9] (a top-down program synthesis in Rust/Python), DeepSynth (general-purpose program synthesizer written in Python).

The main components that are needed to create a program synthesis problem are: grammar (rules that define the search space), search procedure (bottom-up, top-down, etc.), tests (examples that the target program must satisfy). There may also be constraints (rules that restrict the search space further) defined, but they are not necessary for creating such a problem.

For example, the network in Figure 1 can be modeled in the following way (this is a simplified version that is meant to help understand the concept, Richard Wijers [5] synthesizing section provides a more in-depth breakdown), using a Context Free Grammar (CFG):

$$
\begin{array}{lcll}
\mathcal{N} & ::= & \mathcal{N}, (\mathcal{R}, r) \mid (\mathcal{R}, r) & \textit{Network} \\
\mathcal{R} & ::= & \mathcal{L} \to \mathcal{L} & \textit{Reaction} \\
\mathcal{L} & ::= & \mathcal{L} + \mathcal{M} \mid \mathcal{M} & \textit{Molecule list} \\
\mathcal{M} & ::= & \mathcal{M} + \mathcal{A} \mid \mathcal{A} & \textit{Molecule} \\
\mathcal{A} & ::= & \texttt{H} \mid \texttt{O} \mid \texttt{N} \mid \cdots & \textit{Atom}
\end{array}
$$

Grammar 1: Simplified CFG of the base synthesizer, non-capital r refers to the rate at which the reaction occurs.

The type of search procedure can vastly change how the synthesizer behaves in terms of pruning and efficiency. The one that this paper focuses on is a top-down approach, where a root node is used as a starting point and then nodes are unraveled recursively downwards (in other words, going from 'top' to 'down') based on a set of rules. It offers simplicity at the cost of efficiency, in contrast to a bottom-up approach, where partial trees are built and later combined with each other, which can more efficiently prune and sort with constraints and heuristics.

Lastly, there are certain constraints that may be added to this procedure in order to decrease the search space. For example, one of the most common constraints in this kind of procedure is the ordering constraint, which focuses on removing duplicates based on similarities, such as having the same structure but different order, circularity etc.

2.3 Reaction templates

Reaction templates describe the structure of similar reactions in an abstract form, often with the intent of generalizing certain properties from these classes of reactions.

There are many ways to define these reaction templates, as there are existing databases with various types of templates, in different notations and ways of abstracting. In this paper, the main focus is on general definition of reaction templates, often considered as fundamental for describing simple reactions: addition (synthesis/combination), decomposition, single replacement, double replacement and oxidation (combustion); which can be seen in Figure 2.

Figure 2: The 5 most general types of reaction templates, taken from Muhammad Arif Malik's Introduction to General Chemistry. The capital letters (A-D) represent species and capital letter 'O' represents the oxygen molecule.

There are other ways of defining and using reaction templates, as it can be seen in various SMILES[10] notation extraction tools and databases [11], which focus on inferring more fine transformations at a molecular level, rather than the structural reaction information that the general ones provide.

2.4 Related Work

In Coley et al. [12], reaction templates are being used as a way to derive molecules further needed in the synthesis of reactions. More specifically, a training data set of 40000 reactions is used to derive molecule precursors through the process of cleaving (removing) a single bond. Therefore, this work focuses more on deriving molecules based on various similarity methods in the context of reaction templates, whereas this paper analyses a smaller subset of common general reaction templates, with more focus on imposing constraints based on reaction structure.

3 Problem Definition

In the real world, more often than not, we are working with incomplete data, also known as a partially observable problem. When experimenting with reactions, scientists often encounter incomplete CRNs, such that the challenge becomes: how does one infer the missing species in order to obtain the complete CRN, in a plausible way? This problem can be formally stated as follows:

Starting with:

- A set of observed molecules $M_1 = \{m_1, ..., m_n\}$.
- Corresponding concentration measurements over time $C_i(t)$ for each $m_i \in M_1$.

Obtain (ideally in the order of plausibility):

- A set of additional (unobserved) molecules $M_2 = \{m_{n+1}, ...\}$.
- Corresponding concentration measurements over time $C_i(t)$ for each $m_i \in M_2$.
- A CRN, such that simulations of the network reproduce the observed concentration profiles $C_i(t)$ for all $m_i \in M_1, M_2$.

In the end, the complete solution(s) should be available:

- A set of molecules $M = M_1 \cup M_2 = \{m_1, ..., m_n, m_{n+1}, ...\}$
- Corresponding concentration measurements over time $C_i(t)$ for each $m_i \in M$.
- A CRN, such that simulations of the network reproduce the observed concentration profiles $C_i(t)$ for all $m_i \in M$.

4 Method

This section goes over some possible methods and implementations of reaction templates in the context of program synthesizing CRNs. The code can be found in the following Zenodo repository: CRNSynthesizer.

Firstly, constraints at the network level are analyzed, in the form of filters and sorting heuristics based on reaction templates. Then, some changes into the reaction generation/interpreter are considered.

4.1 Network-based Constraints

This method covers a more simpler approach to the problem, by looking at finalized networks and making changes based on filters. One or multiple filters may be applied to a network, such that the final search space will represent all networks which passed at least one filter (in other words, a union of the results). Additionally, these filters are also used as a way to rank networks among each other: all networks receive a score equal to the sum of the filters applied to every reaction the network; a filter applied to a reaction is either 1 if it passes or 0 if it does not match.

As such, for each reaction template, the following aspects about the reactions in the networks are checked in the form of a filter, for which the specifics are available in Table 1:

- Balanced Reaction: the number of atoms on the left (reactants) need to match the number of atoms on the right (products).
- Reaction Structure: reactions need to match the amount of unique reactants and products corresponding to the respective reaction template.
- Any additional requirements for a template that cannot be generalized (i.e. for an oxidation reaction, the oxygen molecule needs to be present in the reactants and isolated).

Filter	**Constraints**
Addition	$A + B \rightarrow AB$ Exactly 2 reactants, 1 product.
Decomposition	$AB \rightarrow A + B$ Exactly 1 reactant, 2 products.
Single-replacement	$AB + C \rightarrow AC + B$ Exactly 2 reactants, 2 products. $\geq$1 reactant and $\geq$1 product atomic or diatomic (X or X_2).
Double-replacement	$AB + CD \rightarrow AD + CB$ Exactly 2 reactants, 2 products. All species have $\geq$2 distinct atom types.
Oxidation	$A + \mathrm{O}_2 \rightarrow AB$ Exactly 2 reactants, 1 product. O_2 must appear as an isolated reactant.

*All filters require a balanced reaction.

Table 1: Network-level reaction filters, describing the constraints imposed on each type of general reaction template.

For the soft constraint (scoring system), all networks are sorted by increasing amount of reactions found in the network, and the filter score is used as a tie-breaker between them.

In the benchmarks, both the hard constraint and soft constraint are tested together, as they are not only similar in terms of how they work, but also complement each other well.

4.2 Reaction Constraint

My second approach is to modify the grammar provided by Herb.jl such that reactions are already created with respect to templates. In the current pipeline, there are three synthesizer levels: molecule, reaction, network. For the purpose of this implementation, we are going to focus on the reaction synthesizer.

The current reaction grammar has the following form:

$\mathcal{R}$	::=	$\mathcal{L} \to \mathcal{L}$	*Reaction*
$\mathcal{L}$	::=	$\mathcal{M} + \mathcal{L} \mid \mathcal{M}$	*Molecule list*
$\mathcal{M}$	::=	$\mathtt{H_2O} \mid \mathtt{O_2} \mid \mathtt{H_2} \mid \cdots$	*Molecule (from synthesis)*

Grammar 2: Reaction grammar (original)

The main issue with this grammar is that it generates all sorts of reactions with no regard to the amount of reactants or products present in the reaction. When comparing all general reaction templates that are being considered, the only constraint that can be imposed on the grammar at the reaction level is to limit the amount of unique molecules present in the reactions:

$\mathcal{R}$	::=	$\mathcal{L} \to \mathcal{L}$	*Reaction*
$\mathcal{L}$	::=	$\mathcal{M} + \mathcal{M} \mid \mathcal{M}$	*Molecule list*
$\mathcal{M}$	::=	$\mathtt{H_2O} \mid \mathtt{O_2} \mid \mathtt{H_2} \mid \cdots$	*Molecule (from synthesis)*

Grammar 3: Reaction grammar (modified)

However, this grammar change cannot be directly implemented into the current code-base, as the 'Molecule' rule is not unique. Herb.jl does provide the ability to limit the unraveling of certain rules through the use of the 'Forbidden()' constraint, such that one could limit the unraveling of the Molecule list rule in the initial grammar more than two times. Using it in this context does not provide the expected result however, as it prevents the generation of reactions with more than 2 molecules per side, unique or not. For example, the following reaction $2\,H_2 + O_2 \to 2\,H_2O$ would not be found, as $2\,H_2 + O_2$ are considered as 3 molecules in the current code-base in terms of grammar rules.

To achieve the same search space as 2nd grammar presented above, a constraint that checks the amount of unique molecules per side of the reaction is added at the level of the reaction interpreter (enumerator). Although this alternative might provide the same results, performance is affected as candidates are generated and then pruned, rather than avoided altogether.

5 Experimental Setup and Results

All implementations mentioned in the Method section are benchmarked against the following 3 experiments: Water Formation (missing O_2), Methane Oxidation (missing O_2 and CO_2), Esterification (missing H_2O, CH_4O and CH_2O_2). All of these problems have a known target network and the main aspects that are going to be emphasized and put into contrast with the baseline synthesizer are:

- Rank of the target network.
- Size of the search space.
- Runtime.

The experiments are run against the following pipeline: Problem → Molecules → Reactions → Networks. This means that after partial data of the problem is considered, a 3 step synthesizing procedure is run in the following order: Molecule Synthesizer, Reaction Synthesizer, Network Synthesizer.

5.1 Water Formation

The target reaction network for Water Formation consists of a single reaction:

$$2\,H_2 + O_2 \xrightarrow{k=0.2} 2\,H_2O \tag{1}$$

The simulation runs over $t \in [0, 10]$ with initial concentrations $[H_2]_0 = 4.0$, $[O_2]_0 = 2.0$, and $[H_2O]_0 = 0.0$. The species observed by the synthesizer (known) are H_2 and H_2O, while O_2 is missing from the input.

Synthesizer settings used: 5 initial molecules, 10 initial reactions, 1000 total networks.

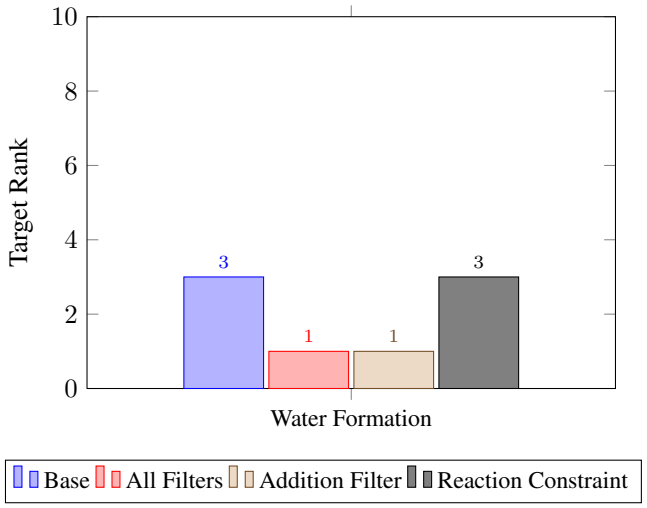

Figure 3: Results for Water Formation benchmark, depicting the target rank for all methods.

The baseline synthesizer obtained a rank of 3 for the target network of Water Formation, whereas both using all filters and only the addition filter yielded rank 1. The reaction constraint did not change the network ranking, obtaining the same rank of 3 as the base, as shown in Figure 3.

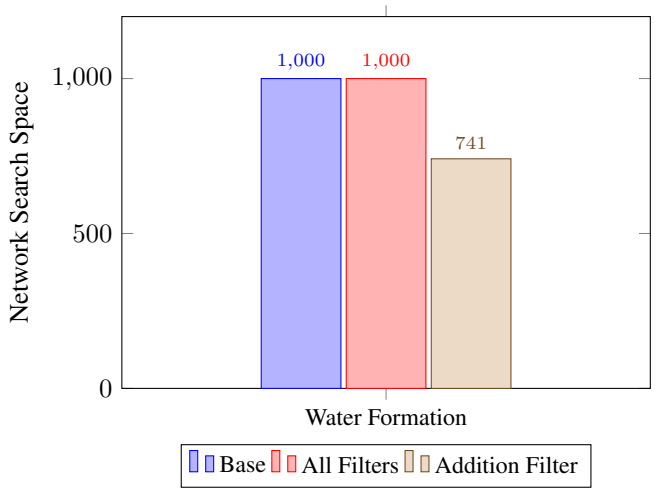

Figure 4: Results for Water Formation benchmark, depicting the network search space for all methods.

In terms of search space, using all filters proves to be not restrictive enough to remove any network, as both no filters (base) and all filters found 1000 networks. Using only the addition filter yielded 741 networks, a 26% decrease in search space, as shown in Figure 4.

5.2 Methane Oxidation (Combustion)

The target reaction network for Methane Oxidation consists of a single combustion reaction:

$$\mathrm{CH_4} + 2\,\mathrm{O_2} \xrightarrow{k=0.2} \mathrm{CO_2} + 2\,\mathrm{H_2O} \tag{2}$$

The simulation runs over $t \in [0, 10]$ with initial concentrations $[\mathrm{CH_4}]_0 = 2.0$, $[\mathrm{O_2}]_0 = 4.0$, $[\mathrm{CO_2}]_0 = 0.0$, and $[\mathrm{H_2O}]_0 = 0.0$. The species observed by the synthesizer (known) are $\mathrm{CH_4}$ and $\mathrm{H_2O}$, while $\mathrm{O_2}$ and $\mathrm{CO_2}$ are missing from the input.

Synthesizer settings used: 10 initial molecules, 100 initial reactions, 1000 total networks.

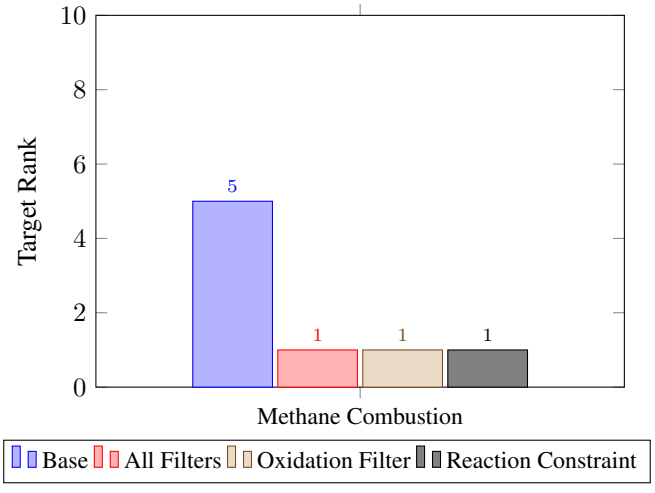

Figure 5: Results for Methane Combustion benchmark, depicting the target rank for all methods.

The baseline synthesizer obtained a rank of 5 for the target network of Methane Oxidation, whereas both using all filters and only the oxidation filter yielded rank 1. The reaction constraint also achieved the same improvement of rank 1, as shown in Figure 5

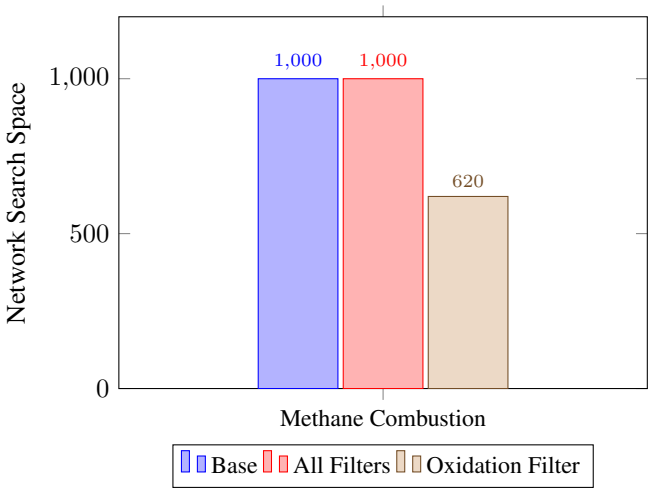

Figure 6: Results for Methane Combustion benchmark, depicting the network search space for all methods.

In terms of search space, using all filters proves to be not restrictive enough to remove any network, as both no filters (base) and all filters found 1000 networks. Using only the oxidation filter yielded 620 networks, a 38% decrease in search space, as shown in Figure 6

5.3 Esterification

The target reaction network for Esterification consists of two reactions:

$$\mathrm{CH_2O_2} + \mathrm{C_2H_6O} \xrightarrow{k=0.2} \mathrm{C_3H_6O_2} + \mathrm{H_2O} \tag{3}$$

$$\mathrm{C_3H_6O_2} + \mathrm{CH_4O} \xrightarrow{k=0.1} \mathrm{C_2H_4O_2} + \mathrm{C_2H_6O} \tag{4}$$

The simulation runs over $t \in [0, 10]$ with initial concentrations $[\mathrm{CH_2O_2}]_0 = 2.0$, $[\mathrm{C_2H_6O}]_0 = 1.0$, $[\mathrm{C_3H_6O_2}]_0 = 0.0$, $[\mathrm{H_2O}]_0 = 0.0$, $[\mathrm{CH_4O}]_0 = 2.0$, and $[\mathrm{C_2H_4O_2}]_0 = 0.0$. The species observed by the synthesizer (known) are $\mathrm{C_2H_6O}$, $\mathrm{C_3H_6O_2}$, and $\mathrm{C_2H_4O_2}$, while $\mathrm{CH_2O_2}$, $\mathrm{H_2O}$, and $\mathrm{CH_4O}$ are missing from the input.

Synthesizer settings used: 70 initial molecules, 56000 initial reactions, 3000 total networks.

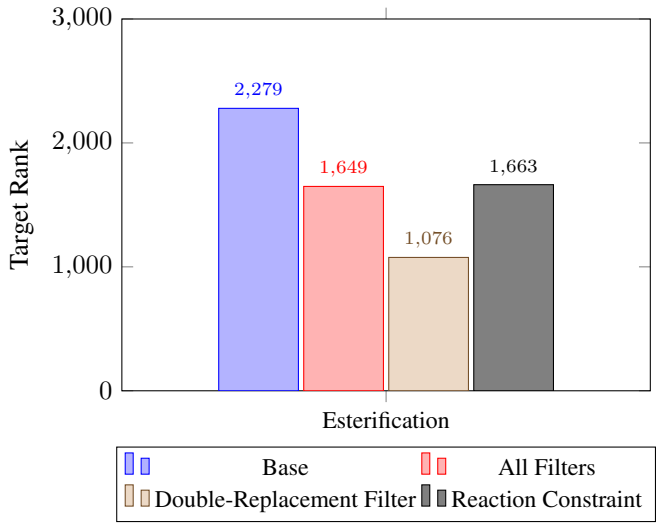

Figure 7: Results for Esterification benchmark, depicting the target rank for all methods.

The baseline synthesizer obtained a rank of 2279 for the target network of Methane Oxidation, whereas using all filters yielded rank 1649, using double-replacement filter yielded rank 1076 and using the reaction constraint yielded rank 1663, as shown in Figure 7.

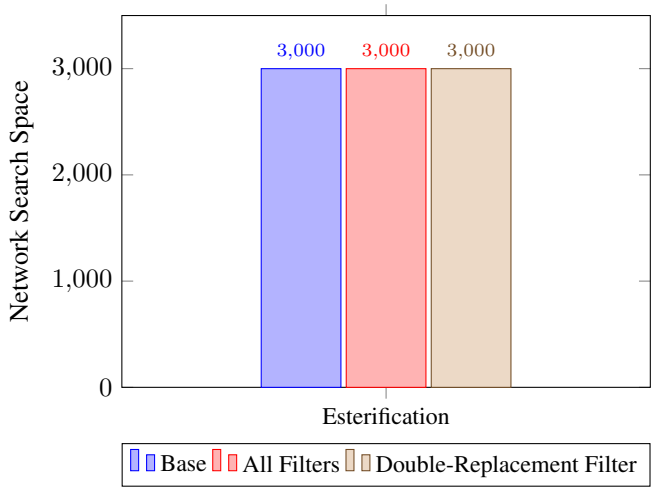

Figure 8: Results for Esterification benchmark, depicting the network search space for all methods.

In terms of search space, using all filters or just the Double-Replacement filter proves to be not restrictive enough, as they both find the same number of networks as the base synthesizer: 3000; as shown in Figure 8.

5.4 Runtime

The benchmarks in the previous subsections are now being compared against the base synthesizer in terms of runtime (execution duration). All of these measurements are taken using ‘@elapsed‘ in-built macro from Julia. In order to avoid inaccuracies created by the macro confusing compile and run time, each benchmark is run twice during the same compilation time: the first time is used as a ‘warm-up‘ in order to compile and cache the synthesizer functions, the second time the ‘@elapsed‘ macro is used to measure runtime.

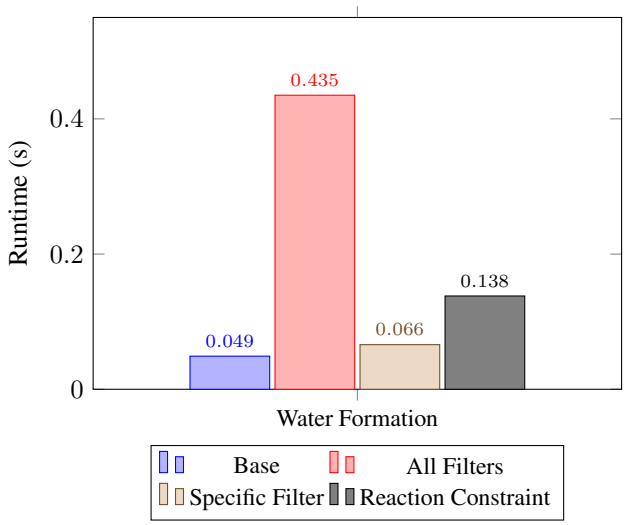

Figure 9: Runtime in seconds for the Water Formation benchmark for all methods.

In the Water Formation benchmark, the baseline synthesizer has an execution time of 0.049 seconds. Using all filters obtains an execution time of 0.435 seconds, a 0.11× speedup, using the addition filter obtains an execution time of 0.066 seconds, a 0.74× speedup, and using the reaction constraint obtains an execution time of 0.138 seconds, a 0.36× speedup.

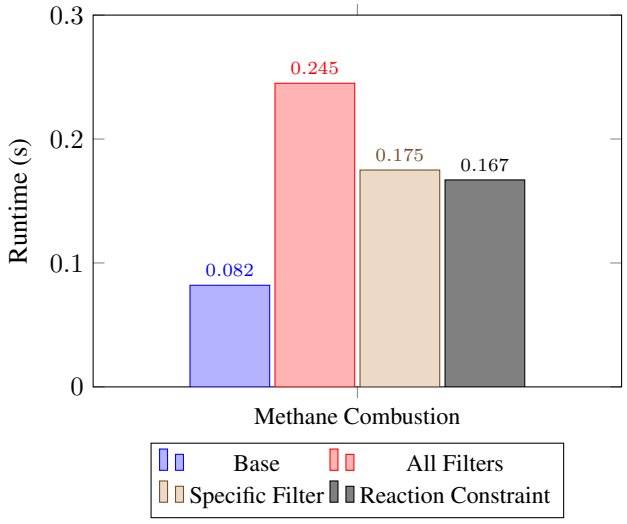

Figure 10: Runtime in seconds for the Methane Combustion benchmark for all methods.

In the Methane Combustion benchmark, the baseline synthesizer has an execution time of 0.082 seconds. Using all filters obtains an execution time of 0.245 seconds, a 0.33×

speedup, using the oxidation filter obtains an execution time of 0.175 seconds, a $0.47\times$ speedup, and using the reaction constraint obtains an execution time of 0.167 seconds, a $0.49\times$ speedup.

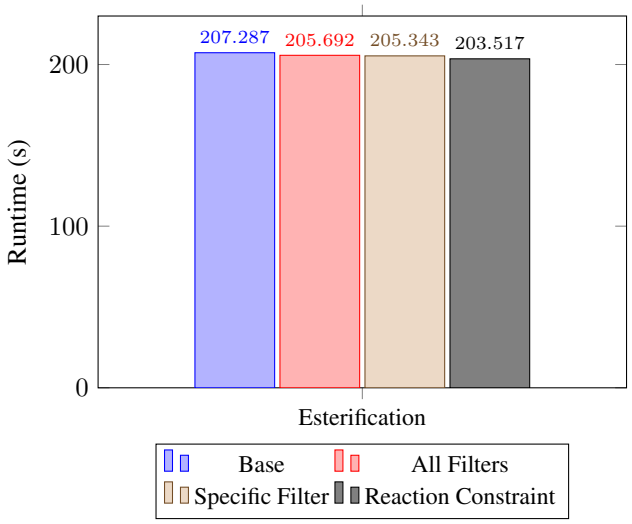

Figure 11: Runtime in seconds for the Esterification benchmark for all methods.

In the Esterification benchmark, the baseline synthesizer has an execution time of 207.287 seconds. Using all filters obtains an execution time of 205.692 seconds, using the double-replacement filter obtains an execution time of 205.343 seconds, and using the reaction constraint obtains an execution time of 203.517 seconds, which all account to a speedup of 1.01-$1.02\times$.

6 Responsible Research

It is important to note that these constraints are a proof of concept, with the goal of trying to get more plausible results. Consequently, if a network is found, there are no guarantees that any of the reactions are possible. Conversely, if a certain network is not found, it does not necessarily mean that it is not a plausible network, but rather that the filters do not cover that kind of reaction or that the settings used are too small.

Moreover, in general working with chemicals can be dangerous, such that any reactions suggested by these implementations should only be attempted by experts in a supervised environment.

In terms of reproducibility, all the code and experiments ran in this paper can be found in the Zenodo repository: CRN-Synthesizer. As far as replicability is concerned, all experiments contain the initial molecules, concentrations and specific ODEs, missing species and target network, as well as the synthesizer settings for the pipeline.

7 Discussion

As mentioned at the start of the paper, the main purpose is to obtain results that are more plausible, either by having a better rank for the target network or by having to check a smaller search space. Therefore, these metrics are analyzed in this section, together with runtime, with the purpose of trying to explain whether these methods are helpful or not.

All benchmarked implementations managed to find the target network within the limitations imposed in each benchmark, as it can be seen in Figures 3, 5 and 7. However, any reaction that does not fall into any of the 5 general reaction templates considered in these methods will not be found in a synthesis procedure. Therefore, assurance of finding the correct target network for any possible CRN reconstruction experiment cannot be guaranteed without having the specific reaction template filter implemented.

For the Network-based constraints, the results show that there was an improvement of target rank for all benchmarks: $3 \rightarrow 1$ for Water Formation (Figure 3), $5 \rightarrow 1$ for Methane Oxidation (Figure 5), $2279 \rightarrow 1649/1076$ for Esterification (Figure 7). This is expected, as reactions that do not match any template were pruned and then the rest were ranked with the scoring based filters.

Using all filters provided no real benefit in terms of search space for any of the benchmarks. This means that the synthesizer already preserves some of the structure required for general reaction templates, at least for the limitations (settings) imposed. Using the respective filter for each CRN does provide some benefit in terms of pruning the search space for both Water Formation (Figure 4) and Methane Oxidation (Figure 6), but not for Esterification (Figure 8). This result is rather unexpected, as it means that all networks being created for Esterification have at least one reaction that matches the double-replacement template, at least in the first 3000 networks.

For the Reaction level constraint, the results show that there is some improvement for the target rank for Methane Oxidation (Figure 5) and Esterification (Figure 7), but not for Water Formation (Figure 3). This can be explained by the fact that the Water Formation CRN is smaller than the rest in terms of unique molecule count, such that the imposed constraint does not get a real chance to prune any candidates before the target is found. However, in the case that more reaction templates are considered, that are not as abstract (general), this constraint can become less effective or even harm correctness.

In terms of runtime, there is noticeable overhead for the smaller benchmarks: Water Formation (Figure 9) and Methane Combustion (Figure 10); especially when all filters are being applied to the networks. However, this overhead completely diminishes in the Esterification benchmark (Figure 11). Considering profiling, this result is expected, as the combinatorial explosion from reaction generation takes over most of the computation.

In terms of bias, it needs to be highlighted the fact that all the settings for the experiments are picked such that all required molecules and reactions are found before the network synthesis procedure.

8 Conclusions and Future Work

8.1 Conclusions

This work covers some of the possible methods that can impose constraints on reconstruction of CRNs, in the con-

text of reaction templates. More explicitly, the most general forms of reaction templates are considered in order to infer constraints on the synthesizer, either at the reaction level or the network level.

All tested methods have shown some improvements, either in terms of improving the target rank or reducing search space, with some diminishing returns for runtime when considering smaller benchmarks. But, considering limitations, such as restricting the problem to only consider general reaction templates, only focusing on constraining the problem rather than possible inference and limited amount of networks benchmarked, the idea of implementing reaction templates into the program synthesis of CRN reconstruction still remains an open challenge.

8.2 Future Work

An important assumption that this study makes is knowing the exact concentrations for each known molecule, which is not necessarily the case in real-world scenarios. Such data is often noisy, which complicates the solving of their respective equations, negatively impacting CRN recovery as a result. As such, future implementations should consider more complex interpretations of species' concentration data, such as one modeled by stochastic differential equations (SDEs with Wiener process), which are already supported by Catalyst.jl.

Another aspect that needs to be noted is that this paper only focused on the 5 most general (abstract) reaction templates, even though there are databases with millions of reaction templates. Therefore, an implementation that considers these databases in their entirety could obtain more interesting results. Even more, exploration of these databases can provide not only constraints imposed on reactions and networks, but also improve molecule generation by only considering the ones present in reaction templates containing the known (input) molecules.

Lastly, the current implementation builds reactions by adding molecules one by one to it, without taking into account the uniqueness of molecules. Consequently, the same reactions are considered multiple times by the reaction interpreter, but in different forms (balancing). One idea that could reduce the search space further would be to only add unique molecules as candidates in a reaction, then attempt to balance the reaction using linear algebra[13].

References

[1] Mingjian Wen, Evan Spotte-Smith, Samuel Blau, Matthew McDermott, Aditi Krishnapriyan, and Kristin Persson. Chemical reaction networks and opportunities for machine learning. *Nature Computational Science*, 3:1–13, 01 2023.

[2] Pavel Loskot, Komlan Atitey, and Lyudmila Mihaylova. Comprehensive review of models and methods for inferences in bio-chemical reaction networks. *Frontiers in Genetics*, Volume 10 - 2019, 2019.

[3] Albert Bruix, Johannes T. Margraf, Mie Andersen, and Karsten Reuter. First-principles-based multiscale modelling of heterogeneous catalysis. *Nature Catalysis*, 2(8):659–670, August 2019. Publisher Copyright: © 2019, Springer Nature Limited. Copyright: Copyright 2019 Elsevier B.V., All rights reserved.

[4] Sumit Gulwani, Oleksandr Polozov, and Rishabh Singh. Program synthesis. *Foundations and Trends in Programming Languages*, 4(1-2):1–119, 07 2017.

[5] R. A. Wijers. Automated discovery of chemical reaction networks using program synthesis. Master's thesis, Delft University of Technology, Delft, The Netherlands, 2025.

[6] Shuan Chen, Juhwan Noh, Jidon Jang, Seongmin Kim, Geun Ho Gu, and Yousung Jung. Reaction templates: Bridging synthesis knowledge and artificial intelligence. *Accounts of Chemical Research*, 57(14):1964–1972, Jul 2024.

[7] Martin Feinberg. Foundations of chemical reaction network theory. *Springer*, 2019.

[8] Tilman Hinnerichs, Reuben Gardos Reid, Jaap de Jong, Bart Swinkels, Pamela Wochner, Nicolae Filat, Tudor Magurescu, Issa Hanou, and Sebastijan Dumancic. Herb.jl: A unifying program synthesis library. *arXiv preprint arXiv:2510.09726*, 2025.

[9] Matthew Bowers, Theo X. Olausson, Lionel Wong, Gabriel Grand, Joshua B. Tenenbaum, Kevin Ellis, and Armando Solar-Lezama. Top-down synthesis for library learning. *Proceedings of the ACM on Programming Languages*, 7(POPL):1182–1213, January 2023.

[10] David Weininger. Smiles, a chemical language and information system. 1. introduction to methodology and encoding rules. *Journal of Chemical Information and Computer Sciences*, 28(1):31–36, 1988.

[11] Pieter P. Plehiers, Guy B. Marin, Christian V. Stevens, and Kevin M. Van Geem. Automated reaction database and reaction network analysis: extraction of reaction templates using cheminformatics. *Journal of Cheminformatics*, 10(1):11, Mar 2018.

[12] Connor W. Coley, Luke Rogers, William H. Green, and Klavs F. Jensen. Computer-assisted retrosynthesis based on molecular similarity. *ACS Central Science*, 3(12):1237–1245, 2017. PMID: 29296663.

[13] Abdelrahim Zabadi and RamizAssaf. From chemistry to linear algebra: Balancing a chemical reaction equation using algebraic approach. *International Journal of Advanced Biotechnology and Research (IJBR)*, Vol-8, Special Issue-Number1:pp– 24, 02 2017.