University of Freiburg

Master Thesis

Atom Mapping of Chemical Reactions via Constraint Programming

thesis submitted in fulfilment of the requirements
for the degree of Master of Science
in the
Chair for Bioinformatics
Department of Computer Science

Done by:
Feras Nahar

Reviewers:
Prof. Dr. Rolf Backofen
Prof. Dr. Christoph Flamm

Supervisor:
Dr. Martin Mann

July 2013

Abstract

A chemical reaction is a process of transforming one set of molecules (educts) into another set (products). In the course of a reaction, chemical bonds which hold the atoms together are redistributed, so that each atom in a reaction educt appears in a specific position of a reaction product. Tracing atoms between educts and products refers to a non-trivial problem in computational chemistry and system biology, namely the "Atom Mapping Problem". Our determination of atom mappings relies on the existence of an imaginary transition state (ITS), in which reacting bonds (formed, broken) are arranged in a cyclic topology. Cyclic mechanisms are very common in chemistry and almost all elementary homovalent and ambivalent reactions feature a cyclic ITS.

The used approach aims at the identification of the cyclic ITS, that imposes additional restrictions on the bijection between educt and product atoms. Once the cyclic ITS is fixed, the overall mapping is easily derived. For this purpose we use Constraint Programming and we show that it is a very promising approach to solve the atom mapping task. The constraint-based model enables the enumeration of atom maps for different cyclic mechanisms and layouts. We present a generic atom mapping framework which based on an encoding that is able to describe different elementary ITS layouts. The generic framework unifies several formulations required to identify different ITS arrangements and it is flexible to incorporate new ones. Our framework also features a method for symmetry exclusion in order to eliminate equivalent mappings and to produce only distinct reaction mechanisms. The performance of the approach is evaluated for a collection of chemical reactions from the KEGG LIGAND database for various ITS cycle layouts. One mapping for most test reactions is located within milliseconds which makes the Constraint Programming approach very appealing in this field.

Zusammenfassung

Eine chemische Reaktion ist ein Transformationsprozess, bei dem Moleküle, die in einer bestimmten Form vorliegen (Edukte), in eine andere Form (Produkte) überführt werden. Während solch einer Reaktion werden chemische Bindungen zwischen den einzelnen Atomen umverteilt, sodass jedes Atom eines Eduktes an einer spezifischen Position eines Produktes wiederzufinden ist. Das sogenannte "Atom Mapping Problem" bezeichnet die Schwierigkeit in der Chemoinformatik und Systembiologie die Position der Atome auf dem Weg vom Edukt zum Produkt zu verfolgen. Bei unserer Untersuchung zur Abbildung von Atomen gehen wir von der Existenz eines imaginären Übergangszustandes (imaginary transition state ITS) mit einer zyklischen Anordnung der umgelagerten Bindungen aus. Diese zyklischen Anordnungen sind in der Chemie weit verbreitet und nahezu alle elementaren homo- und ambivalenten Reaktionen weisen einen zyklischen ITS auf.

Ziel des gewählten Ansatzes ist die Identifikation solches zyklischen ITS, da dieser eine weitere Einschränkung für die Bijektion der Atome von Edukten und Produkten darstellt. Fixiert man den ITS, so ist die vollständige Atomzuordnung einfach zu finden. Zum Detektieren verwenden wir Constraintprogrammierung und wir zeigen, dass diese ein vielversprechender Ansatz zur Lösung des "Atom Mapping Problem" ist. Das Constraintmodell ermöglicht die Abbildung verschiedener gerader oder ungerader zyklischer Mechanismen. In dieser Arbeit führen wir ein generisches Framework zur Zuordnung von Atomen ein, das mittels einer generischen Kodierung verschiedene elementare ITS-Anordnungen beschreiben kann. Es vereint alle notwendigen Implementierungen, die zur Identifizierung von unterschiedlichen ITS-Gebilden notwendig sind. Zudem ist es flexibel genug, um neue Anordnungen mit einzubeziehen. Des Weiteren beinhaltet es eine Methode zum Ausschluss von Symmetrien, damit äquivalente Zuordnungen eliminiert und eindeutige Reaktionsmechanismen identifiziert werden. Die Leistung unseres Ansatzes wurde für verschiedene zyklische ITS-Gebilde mithilfe einiger chemischer Reaktionen aus der KEGG LIGAND Datenbank getestet. Dabei konnte für die meisten Testreaktionen ein Atommapping bereits innerhalb von Millisekunden gefunden werden. Dies spricht für den Einsatz der Constraintprogrammierung auf diesem Forschungsgebiet.

Acknowledgements

I would like to express my deepest thanks to Prof. Dr. Rolf Backofen, Prof. Dr. Christoph Flamm, and Dr. Martin Mann, who gave me the opportunity to work on this interesting topic. Without their constant motivation and great supervision I would not be able to complete this work. A special gratitude deserves my supervisor Dr. Martin Mann for his assistance and for the countless hours he spent in answering my questions. Above all, I would like to thank to my parents and my sister. My mere thanks do not suffice your continuous support and guidance. Finally, I give my thanks to my friends and colleagues and I apologize for not being able to mention them all.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

Chemical reactions are constructed through the propagation of atoms from educt molecules (reaction input) to product molecules (reaction output). Chemical equations are used to represent chemical reactions e.g. $\mathtt{CH_4} + \mathtt{O_2} \rightarrow \mathtt{CO_2} + \mathtt{H_2O}$. As it is shown in the equation, both educts and products are known, however the process of transforming educts to products is unknown. The process of transformation corresponds to changes in the chemical connections, which hold the atoms together, so-called chemical bonds.

Chemical bonds are defined as forces between atoms compounding them in molecular structures. There are different kinds of chemical bonds such as: covalent bonds, Van der Waals forces, hydrogen bonds, and ionic bonds. Here we focus on covalent bonds that are attraction forces between adjacent atoms caused by pairs of valence electrons which are depicted in the figure 1.1. During the course of a chemical reaction, bonds are broken and new bonds are formed yielding products as a result. Due to the ambiguity of this process it is not trivial to trace the atoms in the course of a reaction. Additionally, chemical compounds usually contain a

huge number of atoms which makes a feasible tracing of their positions more awkward. Even reaction databases such as KEGG [7] do not provide the knowledge "which atom in educts is which in products".

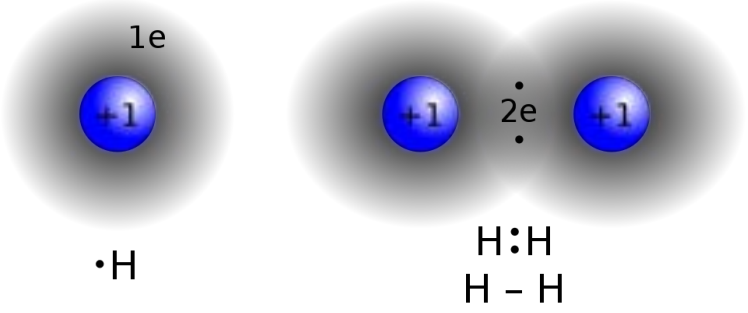

FIGURE 1.1: Valence electron pair (shared electrons) that forms a covalent bond between hydrogen atoms. Source: Wikipedia

The attempts to determine which atoms in educts correspond to which atom in products are called atom mapping. Tracking the position of educt atoms in product molecules has been achieved so far by using isotope labelling experiments. Isotope experiments are based on labelling the educts by means of replacing some certain atoms through their isotopes. Isotopes are chemically identical atoms however they differentiate from the original ones in the number of neutrons. This property makes them recognizable in the products, for instance carbon-12 ^{12}C and carbon-14 ^{14}C are isotopes of the carbon atom. Tracing of isotope positions in the products can be done using nuclear magnetic resonance (NMR) spectra or similar methods [8]. Yet such data is not available for most reactions.

The determination of atom mappings of educt atoms onto product atoms is of high importance and has numerous applications in computational chemistry and system biology. Tracking the atoms during a chemical reaction contributes to identification of chemical changes and thus understanding reaction behaviour/mechanism. Atom maps deliver all required information to infer the mechanism of a chemical reaction without the need for knowledge discovery methods used in reaction databases [9]. Consequently they provide the ability to classify the reactions in terms of the mechanism [4]. Deriving a taxonomy for chemical reactions enables

the chemists to enquiry systematically reaction components assuming a mechanism, so experimental studies and simulation could be drawn based on known mechanisms.

In addtition, atom mapping is used to analyse biochemical pathways in metabolic networks [10, 11]. A metabolic network is a set of interconnecting chemical reactions within the cell typically catalysed by enzymes. It is a very complex framework that consists of a big number of proteins and metabolites involved in lots of reactions. The contained metabolic pathways are required for producing energy, maintaining growth, and reconstruction in cellular processes, hence they are very important for all living organisms. Combining parts of metabolic pathways from different organisms can lead to new ways of deducing how metabolism works and provide new methods to synthesize important and useful compounds. With aid of the reaction mappings, atoms can be traced through metabolic networks to find biologically relevant or realistic pathways from a given start compound to a given target compound [12].

For these reasons, an efficient computation of correct atom mappings is a very important practical problem in computational chemistry. Figure 1.2 exemplifies a reaction atom-atom mapping by assigning numbered labels to each atom.

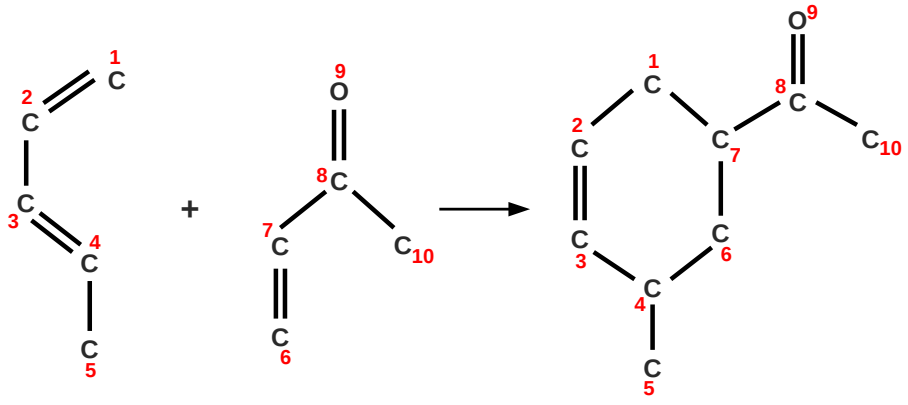

FIGURE 1.2: Atom Mapping: which atoms in educts correspond to which atoms in products. Adapted from Daylight Chemical Information Inc. [1]

1.2 Related Works

Many approaches were proposed so far in order to find atom mappings. It is important to note that the problem of atom mapping is categorized as a computationally hard problem, so it is not trivial to find a feasible solution for it.

Early efforts for finding atom maps were based on an algebraic model to represent chemical reactions that relies on adjacency information of educts and products in form of matrices [13]. For n atoms in a reaction, so-called $n \times n$ bond-electron matrices or simply be-matrices were used to represent the structure of the reaction, such that B, E represent the educts and the products correspondingly. The rows and columns of the be-matrices represent the number and position of the valence electrons of the relevant atoms. Based on be-matrices, the Principle of Minimal Chemical Distance [14] is applied to find chemically meaningful mappings by means of redistribution of the minimum number of valence electrons. The $n \times n$ entries of be-matrices are considered as coordinates of points in $n \times n$ dimensional euclidean space, so that this algebraic logical model is interpreted as a geometric model. The chemical distance denoted $d(B, E)$ is the sum of the absolute values of the differences of the coordinates (entries of B, E matrices) of such that: $d(B, E) = \sum_{ij} |b_{ij} - e_{ij}|$ is minimal. Searching for the minimal distance can be then performed using tree search like Branch-And-Bound.

Recent approaches depend on graph representation [15] of chemical reactions, in which educts and products are expressed using molecule graphs. Chemical atoms are depicted via graph nodes whereas chemical bonds constitute edges connecting adjacent atoms in the molecule graph. The profit of adopting graph representation of reactions is that it allows applying graph algorithms on chemical molecules in order to find atom mappings[1]. The determination of atom mappings is then a solution of a combinatorial optimization problem which maps bijectively all vertices in the educts molecule graph onto corresponding vertices in the products molecule graph.

[1]The most common variants of the atom mapping problem rely on maximum common subgraph isomorphism algorithms.

Molecule graphs of educts and products can be compared in terms of similarity using graph matching or isomorphism algorithms such as Maximum Common Edges Subgraphs (MCES) [16–18]. MCES refers to the largest substructure common to the considered graphs. We say that two graphs G, G' are isomorphic if each node in G corresponds to each node in G' and vice versa and an edge only exists between two nodes in G if an edge exists between the two corresponding nodes in G'. An MCES is a sub-graph consisting of the largest number of common edges in both G and G', in our case in educt and product molecule graphs. If searching for MCES between educt and product graphs results in a match, that means common parts can be mapped on each other. In case of mismatch, edges could not be mapped which means that they were either broken or formed in the course of reaction. Nevertheless MCES graph matching is proven to be an NP-hard problem and fails for certain reactions [19].

Another type of algorithms aims at identification of a reaction center i.e. just those atoms which change their bonding relying on certain energetic criteria [20, 21]. The method for the determination of reacting bonds observes an Imaginary Transition State Energy (ITSE), in which changes to chemical bonds (breakage, formation) occur. The ITSE is determined according to a crude approximation of the reaction energetics, so that the energy of the transition state is minimal. Using minimal energy rule determines the simplest possible reaction center and consequently the atom-atom mapping, since the rest is graph isomorphism.

Certain methods for computing atom mappings reduce the mapping problem to series of chemical substructures until only isomorphic sub-graphs remain [19, 22]. The method in [22] proposes the principle of pattern rearrangement that is applied to enzymatic reactions stored in a metabolic pathway databases such as KEGG [7]. It classifies enzymatic reactions into four rearrangement patters: combination $A + B \leftrightarrow AB$, decomposition $AB \leftrightarrow A + B$, displacement $A + BC \leftrightarrow AC + B$, and exchange $AB + CD \leftrightarrow AD + CB$. The advantage of such partitioning is the availability of fast sub-graph isomorphism algorithms for chemical molecule graphs in polynomial-time.

One of the most recent approaches for the identification of reaction mechanisms through atom maps is based on Integer Linear Optimization (ILP) [23]. This computational method has the advantage of providing atom mappings that are stereochemically consistent. The ILP approach uses on an objective function that minimizes the number of bonds that break or form in order to identify optimal mappings. Additionally, it is capable to exclude equivalent reaction mappings and thus it captures only those mappings that correspond to distinct reaction mechanisms.

In this thesis we exploit the presence of a transition state for finding feasible atom-atom mappings. However we rely on a constraint programming approach to identify this state. When the transition state between educts and products is mapped, then it is easy to extend it to a global mapping for all atoms.

1.3 Contribution

In this thesis we have implemented and extended the constraint-based approach for finding atom maps presented in the articles [2, 5]. Similarly to ITSE-Method mentioned in 1.2, the proposed approach relies on the existence of a reaction center, so-called imaginary cyclic transitional state [24, 25] during the propagation from educts to products. However the detection of the transitional state in our case is (unlike ITSE) energy-independent.

In order to determine those atoms with changing bonds, the approach in [2] uses the fact that almost all chemical reactions in the organic chemistry are described by a cyclic or pseudo-cyclic topology [4, 25]. In a broad sense, the redistribution process of chemical bonds (newly formed bonds, broken bonds) occurs through a transitional state encoded in a cyclic form. Our implementation for finding atom-to-atom mappings focuses on the detection of this cyclic state, called *imaginary transition sub-graph* or simply ITS.

The identification of the ITS imposes to take the specifications of chemical reactions into consideration. So in case of homovalent reactions, where the atomic oxidation number remains unchanged during the reaction, the transition state is elementary. With other words, the imaginary transition graph of such reactions is a single, even-numbered cycle, which enables an alternating arrangement of chemical bonds. Figure 1.3 exhibits the Diels Alder reaction, which features an alternating cyclic ITS. It illustrates bond order changes by ± 1 along the cycle. Once the ITS which connects the educts and products is fixed, the rest of atoms not participating in the ITS can be mapped to each other using a graph isomorphism procedure.

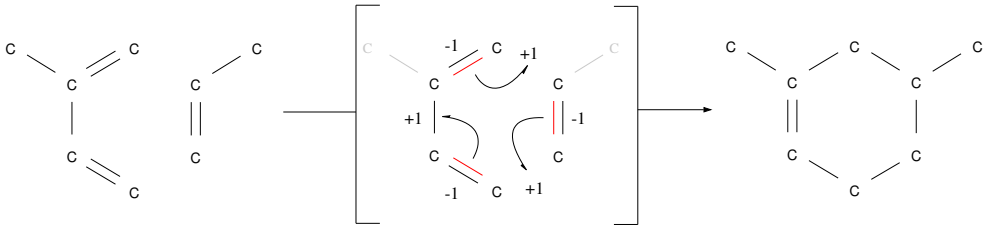

FIGURE 1.3: Example of a Diels-Alder reaction. The ITS is an alternating cycle structure defined by the bonds that are broken (in red) and the bonds that are newly formed. Source: Atom Mapping with Constraint Programming [2].

In this thesis we consider the transition state of elementary reactions (both even and odd), which is represented as a single connected cycle. However non-elementary reactions exhibit complex transition states [25]. The ITS of such reactions is composed of two of more elementary ITSs, which is beyond the scope of this thesis.

Due to the fact that a single chemical reaction can have one or more potential mechanisms of different cycle sizes and layouts, we had to deal in this thesis with three variants of the atom mapping problem:

1. **Decision Problem:** Whether or not there is an atom mapping with associated cyclic ITS of length k.

2. **Optimization Problem:** Find an atom mapping associated with minimal length k of an ITS.

3. **Enumeration Problem:** Find all mappings associated with an ITS of length k.

Providing that the ITS must be an alternating cycle, the basic atom mapping model presented in [2] includes only elementary homovalent reactions. In practice, there exist many more ITS layouts beside the homovalence scheme, so we generalized the approach to incorporate a wider spectrum of chemical reactions, namely elementary ambivalent reactions. Such reactions usually include charged atoms and have an odd-cycled reaction center. For implementing the proposed ITS-centered atom mapping, we exploit the benefits of the constraint programming paradigm in terms of the efficiency in solving such combinatorial problems and its declarative nature that simplifies the modelling of such tasks. The atom mapping problem is then formulated as constraint satisfaction problem to find cyclic ITS candidates for different cycle sizes.

We proposed possible optimizations to make the computation of atom mappings more feasible. This enables to infer to what extent there is a correlation between the performance and the atom number. Furthermore we introduce a method for symmetry exclusion. It eliminates equivalent reaction mappings, so that only distinct mappings are reported as final solution.

The whole approach for both even/odd ring layouts and the corresponding optimizations are then integrated in a generic atom mapping framework. The generic framework provides a universal encoding of elementary ITS rings, which is easily extendible to new ITS patterns. At the end, we provide an evaluation regarding performance and chemical correctness by testing known chemical reactions from the KEGG LIGAND database [7].

1.4 Outline of Thesis

The thesis is organized as follows: chapter 1 introduces the concept of atom mapping of chemical reactions and gives an overview to the existing approaches. The

ITS-centered model for identifying atom maps is presented in chapter 2. Chapter 3 concerns about the realization of the problem using constraint programming. Here the formal model for elementary homovalent reactions from chapter 2 is expressed in terms of combinatorial constraints with corresponding optimizations. Furthermore we extend the constraint model to incorporate elementary ambivalent reactions with odd ITS cycles. Chapter 4 outlines the generic atom mapping framework with implementation details. The following chapter 5 provides an evaluation of the approach by testing various chemical reactions. Finally, chapter 6 contains conclusions about the acquired results as well as future work.

Chapter 2

Reaction Atom-Atom Mapping

This chapter introduces a formal definition of the atom mapping problem as stated in [2]. It presents fundamental concepts such as: molecule graph representation of chemical reactions, adjacency matrices of these molecule graphs, and the imaginary transition subgraph (ITS). For now, the considered atom mapping is restricted to elementary homovalent chemical reactions. At the end we give a quick look on a another type of chemical reactions so-called ambivalent reactions which require special treatment.

2.1 Chemical Reactions as Molecule Graphs

A chemical reaction is a redistribution of valence electrons in educts to produce products. During a chemical reaction the atoms and the total number of valence electrons remain the same. Finding atom mappings between educts and products requires feasible representation of the molecules.

In our case both educt and product molecules are represented as a single, undirected graph by a set of vertices V and a set of edges $E = \{\{v, v'\}|v, v' \in V\}$. The educt (input) graph is denoted by $I = (V_I, E_I)$ and $O = (V_O, E_O)$ for the product (output) graph. The molecule graphs are not necessarily connected, since each single graph can comprise one or more molecules. Atoms within a single molecule

are connected forming the molecule itself as a connected component. Atoms correspond to vertices in the molecule graph and each vertex is labelled with the respective atom type $l(x)$. According to mass conservation principle, the number of atoms in educt and product graphs should be the same i.e. $|V_I| = |V_O|$.

Covalent bonds between atoms correspond to edges in the molecule graphs. At this place, a multi-graph representation might be used, since each bonding electron pair between two atoms can be considered as an edge in the graph, whereas non-bonding electrons[1] conform to loops in the multi-graph structure. When considering atom mapping as constraint satisfaction problem (chapter 3), it will be more suitable to maintain a single graph structure, in which each edge $\{v, v'\} \in E_I \cup E_O$ is labelled with its bond order i.e. two vertices are connected via a single edge labelled with the number of valence electron pairs[2]: 1, 2, or 3 pairs of valence electrons respectively. The following figure 2.1 displays a molecule graph representation of the previously mentioned Diels-Alder reaction from figure 1.3.

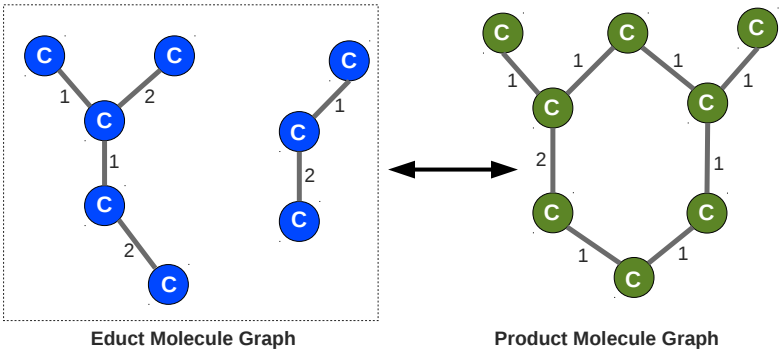

FIGURE 2.1: Educt and product representations through a single undirected (unconnected on the left) molecule graph for each. Edge weights stand for the bonds order.

The adjacency information of the educt graph and the product graph are encoded in two matrices $\mathcal{I}$ and $\mathcal{O}$ respectively. Each entry of $\mathcal{I}_{v,v'}$ and $\mathcal{O}_{v,v'}$ contains the

[1]Non-bonding electron pairs do not contribute to the formation of covalent bonds. They are the left over electrons of the atom and correspond to the atomic oxidation state.

[2]Terminologies: bond order, number of bonding electron pairs, number of valence electron pairs, and edge valence are similar.

bond order of the edge $\{v, v'\}$, such that $\mathcal{I}_{v,v'} \in \{0, 1, 2, 3\}$ and $\mathcal{O}$ as well. Non-bonding electron pairs are stored in the diagonal entries $\mathcal{I}_{v,v}$ and $\mathcal{O}_{v,v}$.

2.2 Atom Mapping Model

We denote function $m : V_I \to V_O$ that maps the vertices of the educts graph onto the vertices of the products graph. Consider a matrix $\mathcal{Q}$ whose rows and columns are indexed by V_I. Finding the corresponding matrix whose rows and columns indexed by V_O is done via composing $\mathcal{Q}$ with the mapping function $\mathcal{Q} \circ m$, such that $\mathcal{Q}_{m(x),m(y)}$ is the required matrix. Hereby $\mathcal{R}^m = \mathcal{O} - (\mathcal{I} \circ m)$ is well defined and encodes the bond electron differences between educt and product.

Definition 2.1. An *atom mapping* is a bijective mapping $m : V_I \to V_O$ such that

1. Atom labels are preserved: $\forall_{x \in V_I} : l(x) = l(m(x))$
2. Total bond orders are preserved: $\mathcal{R}^m \overrightarrow{1} = \overrightarrow{0}$

The *reaction matrix* $\mathcal{R}^m$ encodes the imaginary transition subgraph (ITS[3]) [4, 24]. In other words, it describes the required chemical changes in order to transform reaction educts to reaction products. This definition of m is a slightly more formal version of the Dugundji-Ugi theory [26]. This notation emphasizes the central role of the (not necessarily unique) bijection m. Since I and O are given as fixed input, the imposed bijection uniquely determines $\mathcal{R}^m$. This way the chemical reaction is completely defined using the triple (m, I, O). Therefore the properties of the chemical reaction can be directly associated with the bijective mapping m.

Similarly to educts and products, the ITS encoded in $\mathcal{R}^m$ is represented using a molecule graph $R = (V_R, E_R)$. ITS edges E_R are those edges in I that were removed in O (loss of bonding electrons) and the edges of O which did not exist in I (gain of bonding electrons) i.e. $I_{v,v'} \neq O_{m(v),v(v')} \to \mathcal{R}^m_{v,v'} \neq 0$. The set of atom vertices $V_R \subseteq V_O$ cover all vertices with at least one adjacent edge in E_R. The

[3]ITS is named either imaginary transition state or imaginary transition subgraph.

label of each edge $\{v, v'\} \in E_R$ corresponds to the change in bond order $\mathcal{R}^m_{v,v'} \neq 0$. That means when an edge is removed, the change in bonds order corresponds to the value $\mathcal{R}^m_{v,v'} = -1$, whereby the bonds order is assigned the value $\mathcal{R}^m_{v,v'} = +1$ in case of a newly constructed edge. It is important to know the existence of an atom mapping m does not imply that its matrix $\mathcal{R}^m$ represents a chemically plausible imaginary transition state.

Consider two edges $\{v, v'\}, \{v', v''\} \in E_R$ in R. They are called alternating if $\mathcal{R}^m_{v,v'} \neq 0$ and $\mathcal{R}^m_{v,v'} + \mathcal{R}^m_{v',v''} = 0$ i.e. their values conform to bond-breaking and bond-formation or vice versa in the underlying ITS. We say that R encodes a *simple cycle* of size $k > 2$ when there exists a sequence of vertices $(v_1, v_2, \ldots, v_k, v_1)$ with $v_i \in V_R$, $\{v_i, v_{i+1}\} \in E_R$, $\{v_k, v_1\} \in E_R$, and $\forall i < j \leq k : v_i \neq v_j$. Moreover a simple cycle is called alternating if all successive edges $\{v_i, v_{i+1}\}$ as well as the ring closure $\{v_2, v_1\}\{v_1, v_k\}$ are alternating.

Definition 2.2. An atom map m is *homovalent* if $\mathcal{R}^m_{v,v} = 0$ for all $v \in V_R$. In other words the oxidation number of atoms does not change during the reaction.

Definition 2.3. A homovalent reaction is *elementary* if its ITS is a simple alternating cycle. So $\mathcal{R}^m_{v,v'} \in \{-1, 0, +1\}$ holds for all elementary homovalent reactions.

2.3 Homovalent vs. Ambivalent Reactions

In case of homovalent reactions, atoms do not change valence, so the diagonal entries representing non-bonding electrons of I, O matrices and accordingly R matrix are null. It is important to note that an atom mapping with elementary homovalent ITS can not be found for all chemical reactions [27].

Ambivalent reactions do not admit homovalent requirements. In this type of reactions one or more atoms change their valences via oxidation processes resulting in unshared (delocalized) electrons. While homovalent reactions involve cycles with an even number of atoms, ambivalent reactions feature usually rings with an odd number of atoms due to unshared electrons. The case of odd cycles might cover

an edge which does not alternate with the previous one or with the next. In order to adapt reactions with redox[4] (charged) atoms, the model of elementary homovalent atom mapping has to be extended. Fig. 2.2 shows an ambivalent chemical reaction and how it can be extended to a simple alternating cycle ITS. The il-

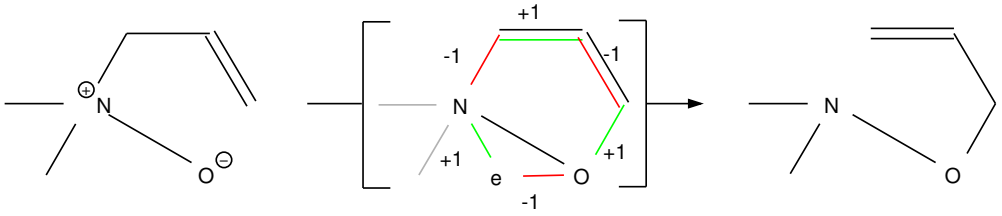

FIGURE 2.2: The Meisenheimer rearrangement [3] transforms nitroxides to hydroxylamines. Source: Atom Mapping with Constraint Programming [2].

lustrated reaction still shows a cyclic ITS with alternating bond electron changes for all but one bond. In order to meet the required cyclic alternating structure, the representation of the graph must be extended. A possible modification can be done by adding a virtual electron node (e^-) at the oxygen. Therefore we need to incorporate an additional "charge separation" rule, so that an electron can have a positive charge (in this case at the nitrogen in the product) to annihilate. Nevertheless, this electron addition would disrupt the bijectivity. Atom mapping of ambivalent reactions is discussed in section 3.4.

Finally, given a straightforward encoding of molecular graphs in terms of vertex indices, atom labels, and adjacency information, the atom mapping problem is naturally open to be treated as a constraint satisfaction problem with finite integer domains.

[4]Redox stands for reduction-oxidation process in which atoms change their oxidation number.

Chapter 3

Atom Mapping as Constraint Satisfaction Problem

Our approach for solving the atom mapping problem is to consider it essentially as a constraint satisfaction problem. The chapter begins with a short overview about Constraint Programming. Afterwards we formulate a basic CSP for the identification of the cyclic ITS for elementary homovalent reactions. This is followed by a description of the graph matching approach to compute the overall atom mapping. In addition, the current chapter provides optimization possibilities of the constraint-based mapping in form of independent CSPs together with an evaluation of their outcomes. At the end, we extend the current CSP to involve elementary ambivalent reactions with respect to different ITS layouts.

3.1 Constraint Programming Overview

Constraint Programming (CP) is a programming technique which uses constraints to describe a problem. A constraint can be understood as a condition to be fulfilled in order to find a solution for the underlying problem. Constraints are denoted as relations between variables and they vary regarding the described problem, for instance, $x \geq y$ or $y = \sum_{i=1}^{n} x_i$. Problems which are declared as constraints and

solved using constraint programming techniques are called *Constraint Satisfaction Problems* or simply CSPs.

Definition 3.1. Bartak[28] defines *Constraint Satisfaction Problem* as follows:

- a set of variables $X = \{X_1, ..., X_n\}$
- for each variable X_i, a finite set D_i of possible values (its domain).
- a set of constraints restricting the values that the variables can take.

Given a function $f_A : X \longrightarrow D$ and the overall domain of all CSP variables $D = D_1 \cup D_2 \cup ... \cup D_n$, we say that a variable $X_i \in X$ is assigned when $f_A(X_i) \in D_i$. A solution for a CSP is found, when each variable is assigned a value from its domain, so that all constraints are satisfied. It might be required (depends on the problem) to find all solutions or only one solution.

The constraints can involve an arbitrary number n of variables. We distinguish between three kinds of constraints: *unary*, *binary* and *n-ary* constraints. An unary constraint is a relation on a single variable e.g. $X \leq 3$, so only the domain of the variable matters. While binary constraints are posted on two variables such as $X \neq Y$, so here the domains of two variables are of concern, so that they both satisfy the condition. The case of n-ary constraints covers a relation, in which all domains of n variables have to be considered. The constraints used to model constraint-based atom mapping are combinatorial binary and n-ary constraints (chapter 3.2.1).

Modelling and solving tasks using constraint satisfaction approaches have specific advantages over modelling a problem say, as a mathematical programming problem. The Formulation of problems as CSPs is easier than expressing them for instance using linear inequalities. Moreover, a CSP is closer to the original problem and easier to maintain due to its declarative nature. CP also allows to apply quick search algorithms for finding solutions efficiently, comparing with common programming methods. It is important to note that CP is used to solve combinatorial NP-hard problems such train and aircraft scheduling, staff planning,..,etc.

3.1.1 Constraints Propagation and Search

The approach of constraint programming separates between modelling the problem and solving it. Each problem is declared in terms of variables and constraints and then is passed to a constraint-solver which is responsible for finding a solution, if any. The constraint-solver is responsible for restricting the domain values of the variables to those which do not violate the posted constraints. The process of pruning domain values, so that they satisfy the required constraints is called *constraints propagation*. Example 3.1 shows a set of acceptable domain values for minus and order constraints after propagation.

Example 3.1. Given set of variables $X = \{A, B, C\}$ and set of domain values $D = \{0, 1, 2, 3, 4, 5\}$. Then the propagation of the order constraints $A < B$ and $B < C$ results for example in the following domains: $D_A = \{0..3\}$, $D_B = \{1..4\}$, and $D_C = \{2..5\}$. One correct assignment of the the variables A, B, and C from the above domains is: $f_A(A) = 2$, $f_B(B) = 4$, and $f_C(C) = 5$.

A propagation-based constraint solver performs search as well in order to get a solution i.e. a complete assignment fulfilling all constraints. A general CP search strategy depends on splitting the problem space into smaller problems and then solves them recursively. The used recursion scheme maintains usually backtracking search by means of exploring a tree of solution-candidates and traversing it recursively to determine valid solutions. A CSP can use different search algorithms such as Depth-First-Search (DFS) or Branch-And-Bound (BAB) which is usually applied in *Constraint Optimization Problems* (COPs)[1].

In this thesis we used the Gecode [29] as constraint solver. Gecode is an open source C++ library featuring a generic constraint-solving framework. It comes with a range of efficient propagators that were used to implement some of the constraint posed by the approach. Furthermore, it provides a framework for defining and implementing problem-specific constraints and propagation strategies, which were used to model the atom mapping problem.

[1] A COP is a CSP with a weight function controlling the quality of solutions in order to find the optimal one.

3.2 Atom Mapping of Homovalent Reactions

The approach in [2, 5] relies on the constraint programming paradigm for finding a solution for the atom mapping problem. To simplify the representation, we focus in following CSP formulation on elementary homovalent reactions. The generalization of the approach is discussed later in 3.4. In this thesis, we have implemented and extended the approach in [2, 5]. Additionally, we proposed improvements for the constraint-based model and discuss their impacts. Each improvement is formulated as a separate CSP.

The main idea as stated in 2.2 is to determine the alternating ITS ring. Once the ITS has been identified, the overall atom mapping is easily derived. A fast graph matching approach is used subsequently to extend each ITS to a global atom mapping. Identifying the desirable cyclic alternating ITS implies posting specific combinatorial constraints to be fulfilled by the mapping of educts to products. Thus the whole attempt for finding an atom mapping is modelled as CSP. Because of the focus on homovalent reactions that show the alternating cycle condition of the ITS, only cycles with even numbers of atoms are considered in following formulation. In practice, elementary homovalent reactions involves $|V_R| = 4$, 6, or 8 atoms in their ITS rings [30].

3.2.1 Basic CSP Formulation

A CSP for an ITS cycle of size $|V_R| = k$ is given by the triple (X, D, C) defining the set of variables X representing the atoms (the nodes of the educt/product graphs), corresponding finite domains (educt/product atoms), and the set of constraints C to be satisfied by any solution.

We construct an explicit encoding of the atom mapping using k nodes of educt and product molecule graphs involved the ring. In a broad sense, the following encoding describes the identification of the educt ITS subgraph and the corresponding ITS subgraph in the product, so it does not directly encode the overall atom mapping.

We introduce a set of node variables $\{X_1^I, \ldots, X_k^I\}$ in the educt I and another set $\{X_1^O, \ldots, X_k^O\}$ for the mapped nodes in the product O, i.e. $X = \{X_1^I, \ldots, X_k^I\} \cup \{X_1^O, \ldots, X_k^O\}$ with domains[2] $D_i^I = V_I$ and $D_i^O = V_O$. The required constraints for the atom mapping CSP are the following:

1. **Bijective Mapping:** All variables must be assigned distinct values in order to ensure bijective mapping, i.e. $\forall i \neq j : X_i^I \neq X_j^I$ and $\forall i \neq j : X_i^O \neq X_j^O$.

2. **Label Preservation:** An atom label is given as $l(x)$ for $x \in V_I \cup V_O$. The corresponding atom labels between educts and products must be equal $l(X_i^I) = l(X_i^O)$, i.e. we have to enforce $\forall e \in D_i^I : \exists p \in D_i^O : l(e) = l(p)$ as well as $\forall p \in D_i^O : \exists e \in D_i^I : l(p) = l(e)$.

3. **Homovalence:** The number of non-bonding electron pairs in the homovalent reactions does not change during the reaction. Consequently the differences between all combinations of the diagonal variables in the matrices $\mathcal{I}$, $\mathcal{O}$ are zero, so $(\mathcal{I}_{X_i^I, X_i^I} - \mathcal{O}_{X_i^O, X_i^O}) = 0$.

4. **Alternating Cycle:** This constraint represents the alternating cycle structure of the ITS, i.e. for the sequence of bonds with indices 1-2-..-k-1. For all ring pair indices (i, j), it is required that pairs with even index i to correspond the formation of a bond, so we enforce $(\mathcal{O}_{X_i^O, X_j^O} - \mathcal{I}_{X_i^I, X_j^I}) = 1$. While all odd indices i are bond breaking $(\mathcal{O}_{X_i^O, X_j^O} - \mathcal{I}_{X_i^I, X_j^I}) = -1$. For example the ring pair $(1, 2)$ in the bond sequence corresponds to bond breaking, which means $\mathcal{O}_{X_1^O, X_2^O} - \mathcal{I}_{X_1^I, X_2^I} = -1$.

5. **Edge Degree:** The alternating cycle condition enforces that each atom can loose or gain at most one edge in the course of a reaction (Fig. 1.3). Therefore we restrict the formation of new edges and the breaking of old ones to be at most one by $|degree(X_i^I) - degree(X_i^O)| \leq 1$; where $degree(v)$ is the number of out-edges of vertex v.

[2] Although the domains above correspond to vertices $v \in V_I \cup V_O$, they are easily represented in Gecode as integer values via a vertex numbering, in order to apply Gecode propagators.

6. **Input Order:** This constraint is posted on educt variables to exclude symmetric solutions, that arise from ITS rotation symmetries. In order to eliminate symmetric matches of the ITS graph on itself, we tie the the smallest cycle node to the first educt variable X_1^I and post an index order on the educt vertices i.e. $(\forall i > 1 : X_1^I < X_i^I)$; where $X_i < X_j$ denotes $\exists(x, y) \in D_i \times D_j : x < y$. This way we can fix the direction of the ring. For more information on symmetry exclusion see 4.4.

The basic CSP is outlined in the following algorithm 1 using integer domains :

Algorithm 1 Identification of even ITS for elementary homovalent reactions

Require: eduAtoms, proAtoms are arrays of size $k \in \{4, 6, 8\}$
dom(eduAtoms, 1, $|V_I|$), dom(proAtoms, 1, $|V_O|$)
Ensure: even ITS of size k

▷ Bijective mapping between educt and product
distinct(eduAtoms)
distinct(proAtoms)
▷ order constraint on ITS educt variables to void rotation-symmetric solutions
for $i = 2 \rightarrow k$ **do**
  rel(eduAtoms[1],LE,eduAtoms[i]) ▷ LE stands for less than in gecode
end for
▷ Setting label preservation, homovalence, and edge degree constraints
for $i = 1 \rightarrow k$ **do**
  preserveLabel(eduAtoms[i],proAtoms[i])
  homovalent(eduAtoms[i],proAtoms[i])
  edgeDegree(eduAtoms[i],proAtoms[i],1) ▷ Loss or gain at most one bond
end for
▷ Ensure alternating cycle structure of the ITS in the mapping
for $i = 1 \rightarrow k - 1$ **do**
  if $(i \, mod \, 2 = 0)$ **then** ▷ Bond formation in case of even indices
    alternateCycle(eduAtoms[i],eduAtoms[i+1],proAtoms[i],proAtoms[i+1]+1)
  else ▷ Bond breakage in case of odd indices
    alternateCycle(eduAtoms[i],eduAtoms[i+1],proAtoms[i],proAtoms[i+1],-1)
  end if
end for
▷ Alternating cycle for ring closure
if $(k \, mod \, 2 = 0)$ **then**
  alternateCycle(eduAtoms[k],eduAtoms[1],proAtoms[k],proAtoms[1],+1)
else
  alternateCycle(eduAtoms[k],eduAtoms[1],proAtoms[k],proAtoms[1],-1)
end if

Note that the mentioned constraints were implemented using Gecode propagators. All new propagators are binary (two variables combinatoric), excepts the alternating cycle constraint which propagates on four variables.

The mapping of the cycle is determined, once the constraints above are met. It is still open to map the rest of the atoms which are not members of the ITS to ensure the chemical correctness of the found cycle. In the following section we discuss how to extend an ITS candidate to a complete atom mapping.

3.2.2 Overall Mapping Procedure

CSP Solutions that fulfil the mentioned constraints are considered as ITS candidates. In order to derive the complete atom mapping, we need to check whether an ITS candidate is chemically valid or not. We say that a CSP solution is valid, when the rest of the atoms not participating in the ITS can be mapped without any further bond formation or breaking. This is checked via standard graph matching approach.

To compute the overall atom mapping we enumerate solution-candidates for all possible ITS ring sizes $k \in \{4, 6, 8\}$. A CSP solution is denoted with a_i^I and a_i^O the assigned values of the variables X_i^I and X_i^O respectively. For each ITS candidate the procedure of graph matching is applied in order to detect isomorphism between the graph parts of educts and products, which are outside the ITS ring. Therefore we need to relabel the edges of the ITS ring pairs by assigning them a unique label, so that they can be identified uniquely in both educts and products molecule graphs. For this reason, we derive two new adjacency matrices $\mathcal{I}'$ and $\mathcal{O}'$ from the original matrices $\mathcal{I}$ and $\mathcal{O}$ respectively, as follows: For all ITS ring pairs (i, j) within the ring sequence 1-2-..-k-1, we change the corresponding adjacency information to the unique label using $\mathcal{I}'_{a_i^I, a_j^I} = \mathcal{O}'_{a_i^O, a_j^O} \in \{f, b\}$ declaring if a bond between the mapped ITS nodes is formed (f) or broken (b). All other adjacency entries remain the same as in $\mathcal{I}$ and $\mathcal{O}$.

According to the updated encoding of ITS edges in the new adjacency matrices $\mathcal{I}'$ and $\mathcal{O}'$, the identification of the overall atom mapping m reduces to the graph isomorphism problem based on $\mathcal{I}'$ and $\mathcal{O}'$. If the graph matching yields an exact mapping of $\mathcal{I}'$ onto $\mathcal{O}'$, then the found atom mapping of the underlying homovalent elementary reaction is a valid mapping, otherwise it is invalid despite the solution satisfying the constraints. The graph matching is done using the efficient VF-2 algorithm [31] which is among the fastest available [32].

3.3 Atom Mapping Approach Extension

Most chemical reactions in the organic chemistry consist of a big number of atoms. Since the options of atom mapping are correlated with the number of atoms within a reaction, they increase drastically with the increment in the atoms number. Consequently the problem size of constraint-based atom mapping becomes larger. After implementing the basic CSP 3.2.1 in Gecode, we get the following results when testing the Diels Alder reaction form the figure 1.3:

Overall Solutions	1.948.184
Valid Solutions	1
Propagations	15.208.279
CSP Time	32.52 s
VF-2 Time	355.03 s
Total Time	387.55 s

TABLE 3.1: Results of the basic CSP. Field "Overall Solutions" gives the number of CSP solutions (ITS candidates) tested via VF-2. "Valid Solutions" denotes the number of chemically correct solutions (matched in VF-2) excluding symmetries. "Propagations" represents the number of the constraint propagations within the CSP required to enumerate atom mappings. Timings are given in seconds and correspond to the required time for the CSP formulation, time for the VF-2 graph matching, and the overall time for CSP and VF-2 together.

Considering the Diels Alder reaction, each carbon atom must have four covalent bonds (not illustrated in the figure 1.3 for simplicity). Missing bonds are connections to hydrogen atoms, in this case 14 `C-H` bonds. Even for such a small chemical reaction, an intensive computation is required in order to determine a chemically correct ITS ring.

When observing the results of the table 3.1 above, we notice that out of 1.948.184 ITS candidates there is only one correct solution (chemically valid ITS cycle) and the remaining 1.948.183 ITS-candidates are pseudo-solutions. Pseudo-solutions satisfy the constraints, however they are either not chemically valid i.e. they do not enable an atom mapping over the whole educt and product graphs regarding the graph matching procedure or symmetric. As a result, the VF-2 graph matching procedure in the previous test was unnecessarily 1.948.183 times executed raising the execution time enormously to 387.55 seconds. Additionally we face here the problem of an immense symmetric combinatoric of solutions. The basic CSP results in a large set of equivalent mappings due to the exchange of atoms inside ITS. Thus it is required to reduce the number of pseudo-solutions by incorporating further restrictions into the CSP while identifying the ITS and by excluding symmetric solutions (see 4.4).

In the following sections, we propose extensions of the basic CSP in terms of posting new constraints that allow an efficient enumeration of atom mappings. These extensions aim at the domains restriction of the combinatorial variables and thus the reduction in the number of atom maps produced by the basic CSP. The enhancements are formulated as separate CSPs, namely Minimal Edge Valence of ITS Atoms 3.3.1, Extended CSP 3.3.2, Full CSP 3.3.3 and Edge Valence Conservation CSP 3.3.4. Figure 3.1 shows the hierarchy of the proposed extensions.

Note that all suggested extensions were tested using the Diels Alder reaction (Fig. 1.3), for different reactions see chapter 5 “Tests and Evaluation”.

3.3.1 Minimal Edge Valence of ITS Atoms

We can accelerate the determination of the alternating cyclic structure of the ITS by filtering bond candidates for the ITS. Hereby we incorporate an additional constraint into the basic CSP formulation 3.2.1, which acts as a filter for the minimal required bond order. It says that the minimal edge valence for broken bonds in I and formed bonds in O has to be at least one.

This minimal edge valence propagates on k ITS bonds ensuring that for all ring pair indices (i, j) within a bond sequence 1-2-..-k-1, pairs with even index i in products correspond to $\mathcal{O}_{X_i^O, X_j^O} \geq 1$ in order to result in a bond formation, while pairs with

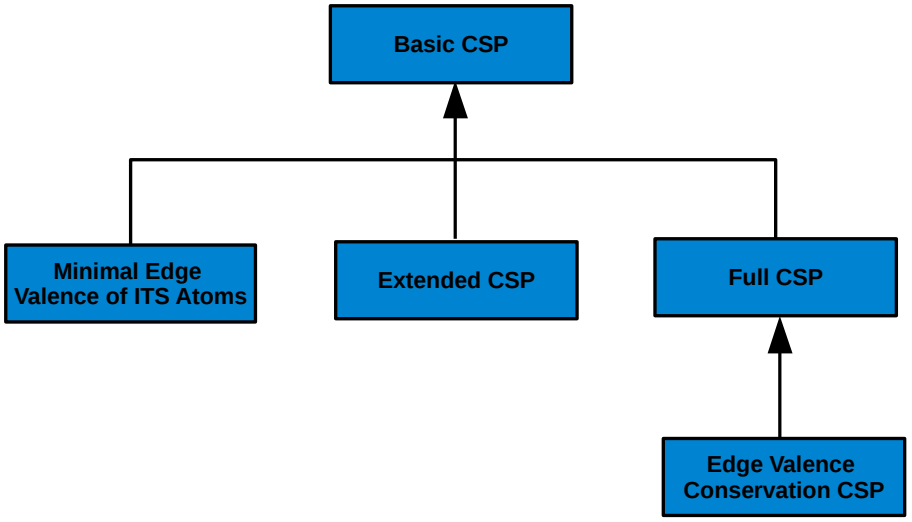

FIGURE 3.1: The hierarchy of the CSP model extensions. Each CSP extends the basic CSP by employing specific constraints to speed up the enumeration of atom mappings.

odd indices i in educts to $\mathcal{I}_{X_i^I, X_j^I} \geq 1$ (bond breakage). In other words, we consider during the determination of the cyclic alternating ITS only those bond pairs in O and I that can result in a bond order change of 1 and -1 respectively. The results of this extensions demonstrated in the table 3.2 show that it does not have an impact on the computation of atom mappings. For Diels Alder reaction (Fig. 1.3), there were no change in terms of the number of ITS candidates and the timings are almost the same. Furthermore, the minimal edge valence enforcement leads to an increase in propagation effort. Nevertheless, this constraint might be useful when testing other reactions.

Overall Solutions	1.948.184
Valid Solutions	1
Propagations	18.087.034
CSP Time	32.05 s
VF-2 Time	353.2 s
Total Time	385.25 s

TABLE 3.2: Results of the addition of "Minimal Edge Valence of ITS Atoms". See table 3.1 for field declarations.

3.3.2 Extended CSP - Precomputation of ITS Members

The central problem in the basic CSP is the production of a huge number of pseudo-solutions, so we focus on the reduction of invalid ITS cycles. In this sense, we can take advantage of the fact that educts and products are given as fixed input. This can be used to determine the ITS candidates more efficiently via suitable precomputations.

To this end, we can compare the graph structure of educt and product molecules to find in advance, before formulating the CSP, a lower bound of atoms (number and type) that will participate in the ITS ring. The central idea is to generate local neighbourhood sets N_I and N_O of all atoms for the educt and product graph, resp., given by

$$N_I = \{\, N(v) \mid v \in V_I \} \text{ with} \tag{3.1}$$

$$N(v) = (\, l(v), \{\, \mathcal{I}_{v,v'} \oplus l(v') \mid \text{where } v \neq v' \in V_I \wedge \mathcal{I}_{v,v'} > 0 \} \,) \tag{3.2}$$

where $N(v)$ is a tuple of the label of atom vertex v and an encoding of the set of all adjacent edges for this vertex. Note, $\oplus$ denotes string concatenation. N_O is derived accordingly. For example, the neighbourhood sets for the Diels Alder reaction from Fig. 1.3 are:

$$\begin{aligned} N_I &= \{\, 2\times(\mathtt{C}, \{\mathtt{1C}\}), 3\times(\mathtt{C}, \{\mathtt{2C}\}), 2\times(\mathtt{C}, \{\mathtt{1C}, \mathtt{2C}\}), (\mathtt{C}, \{\mathtt{1C}, \mathtt{1C}, \mathtt{2C}\}) \,\} \\ N_O &= \{\, 2\times(\mathtt{C}, \{\mathtt{1C}\}), 3\times(\mathtt{C}, \{\mathtt{1C}, \mathtt{1C}\}), (\mathtt{C}, \{\mathtt{1C}, \mathtt{2C}\}), (\mathtt{C}, \{\mathtt{1C}, \mathtt{1C}, \mathtt{1C}\}), \\ &\quad (\mathtt{C}, \{\mathtt{1C}, \mathtt{1C}, \mathtt{2C}\}) \,\} \end{aligned}$$

It is now possible to determine the minimal number of certain atoms that will appear for sure in the ITS. This is achieved via the subtraction $N_I \setminus N_O$. Set difference gives the local neighbourhood that is unique within the educts, which means it has to be changed (formed, broken) in the course of the reaction. Therefore it is guaranteed that a number of atoms of certain type are part of the ITS. In the example this results in $N_I \setminus N_O = \{3\times(\mathtt{C}, \{\mathtt{2C}\}), (\mathtt{C}, \{\mathtt{1C}, \mathtt{2C}\})\}$ revealing that at least 4 C-atoms of two types are ITS members.

Given this information, we formulate an extended version of the basic CSP. and enforce that a valid assignment of the input variables X^I and X^O preserves the ITS neighbourhoods $N_I \setminus N_O$ and $N_O \setminus N_I$, respectively. To minimize propagation effort, this is ensured by an n-ary constraint propagating only after all variables have been assigned to a single value (full assignments).

Employing the neighbourhood constraint to the basic CSP has significantly decreased the solutions to 134 mappings due to the successful precomputation of four ITS-participating carbon atoms in a 6-cycled Diels Alder reaction. As stated in the following table 3.3, the runtime of the overall mapping has been greatly reduced to 1.01 s.

Overall Solutions	134
Valid Solutions	1
Propagations	213.624
CSP Time	0.94 s
VF-2 Time	0.07 s
Total Time	1.01 s

TABLE 3.3: Results of the addition of "Local Neighbourhood Lists" precomputation. See table 3.1 for field declarations.

3.3.3 Full CSP - Involvement of non-ITS Atoms

The basic CSP 3.2.1 describes k atoms in the educts and in the products accordingly. Those k atoms need to satisfy all previously mentioned constraints since they are part of the ITS cycle. However the domains of those k variables are V_I and V_O correspondingly i.e. including all nodes in educts and products graphs.

We can expand the basic CSP to involve the rest of the atoms $n - k$, which do not participate in the ITS. Those atoms do not need to fulfil all ITS constraints, they just need to preserve atom labels, node degree and bond valence information. In other words we have to ensure that non-ITS atoms in both educts and products are mapped properly (conform to each other regarding their labels and edges). In addition to k ITS nodes, the encoding of the full CSP incorporates nodes which are not involved in the ITS as following: $X' = \{X'^I_1, \ldots, X'^I_{n-k}\} \cup \{X'^O_1, \ldots, X'^O_{n-k}\}$ with domains $D'^I_i = V_I$ and $D'^O_i = V_O$. For these variables, we post bijective mapping and atom label preservation from the constraints above. Additionally, we need to preserve the **local adjacency** of

non-ITS atoms nodes, such that $\Big(degree(X_i'^I) = degree(X_i'^O)\Big) \Rightarrow \{v_l | v_l = I_{X_i'^I,l}\} = \{v_r | v_r = O_{X_i'^O,r}\}$ for $1 \leq l \leq r \leq n-k$.

The engagement of non-ITS atoms poses more restrictions on the graph structure to be met by the constraints and results in a reduction of invalid ITS cycles. The number of solutions has been reduced to 1.035.476 in the full CSP, instead of 1.948.184 in the basic variant. The time consumption of graph matching algorithm VF-2 has been reduced as well, due to the decrement in the number of invalid ITS rings. Nevertheless because of extra constraints, the problem size becomes larger increasing this way the propagation as shown in the table 3.4. The number of pseudo-solution is still very big comparing with extend CSP 3.3.2.

Overall Solutions	1.035.476
Valid Solutions	1
Propagations	24.418.582
CSP Time	44.91 s
VF-2 Time	196.82 s
Total Time	241.73 s

TABLE 3.4: Results of the full CSP. See table 3.1 for field declarations.

3.3.4 Full CSP- Edge Valences Conservation

The full CSP 3.3.3 still provides lots of invalid solutions. To reduce their number, it is needed to pose more structural limitations on the molecule graphs with respect to non-ITS nodes. We need to ensure that all edge valences (not only local bond valence information) outside the ITS cycle are conserved. That means for each edge between two non-ITS educt variables $X_i'^I$, $X_j'^I$ with certain bond order, then we want to enforce that corresponding non-ITS product variables $X_i'^O$, $X_j'^O$ are connected via an edge weighted with same bond order as well.

The restriction of graph structure through non-ITS atoms aims at making the educt and product graph parts outside the ITS ring almost isomorphic. Therefore the preservation of **full adjacency** is based on the full CSP and is posted for all possible non-ITS atom pair combinations such that for all $(n-k)^2/2$ pairs, where $i < j : \mathcal{I}_{X_i'^I,X_j'^I} = \mathcal{O}_{X_i'^O,X_j'^O}$. As it is apparent in the table 3.5, this constraint contributes to cutting down the number

of invalid ITS mappings passed to the VF-2 matching step. The implementation of this constraint reduced the number of solutions considerably to 10.508 comparing with 1.035.476 solutions in the full CSP variant.

Overall Solutions	10.508
Valid Solutions	1
Propagations	16.843.168
CSP Time	64.09 s
VF-2 Time	1.93 s
Total Time	66.02 s

TABLE 3.5: Results of the addition of "Edge Valences Conservation" constraint. See table 3.1 for field declarations.

Observing Diels Alder 1.3 reaction, the best optimization so far was the extended CSP 3.3.2, which produces the minimal number of invalid ITS candidates: 134 pseudo-solutions. Nevertheless, it might not always be the case when testing different chemical reaction, as it is discussed in chapter 5 "Tests and Evaluation".

3.4 Atom Mapping of Ambivalent Reactions

The algorithm outline in 3.2.1 and its enhancements enumerate all possible atom maps only for elementary homovalent reactions. Ambivalent reactions (shortly mentioned in 2.3) feature usually an odd-cycled imaginary transition state. So we need to extend the formulation of the atom mapping CSP to incorporate elementary ambivalent reactions. We discuss here the required changes in the formulation of the constraints to allow the identification of odd-cycled mechanisms. Due to ambivalence, odd ITS rings can have different ring layouts, so we need to formulate different CSPs based on the observed ITS layout.

3.4.1 Ambivalent Reactions and Odd ITS Cycles

So far we have investigated homovalent reactions with even-numbered cycle of atoms. However, the homovalence does not always hold i.e. not all chemical reactions maintain constant valences during the transformation from educt into product. One or more

atoms can change their valence by means of gaining or losing non-bonding electrons. This is caused by so-called redox processes which lead to the delocalization of electrons within a certain molecule. Due to the change in non-bonding electrons[3] and thus the atomic oxidation state, such reactions are not homovalent, but ambivalent.

An ambivalent atom with delocalized electrons can be positively or negatively charged and is called ion. In the ambivalent reaction from the following figure 3.2, the unshared electron changes into a bond in case of the oxygen ion $\mathtt{O}^{-1}$. On the other side, positively charged atom causes a bond breakage, since it receives an electron from an adjacent bond ($\mathtt{N}^{+}$ below). Charged atoms are associated with atomic oxidation state changes (or simply charge changes), which indicate the number of transfer electrons that must be added up to the charge on ion (see the figure below) to turn into a neutral atom. Thus the charge change describes the gain or loss of electrons for the ambivalent atoms required to change them to neutral elements. For this odd arrangement of the ITS that contains two oppositely charged ions, we will formulate an according CSP in the section 3.4.2.1.

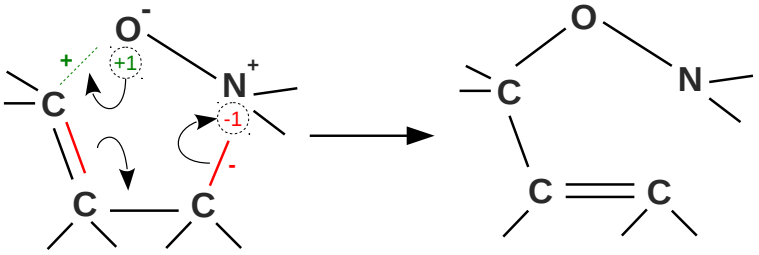

FIGURE 3.2: The Meisenheimer rearrangement [3], adapted from [2]. Red bond are broken, green dotted bond is formed. The numbers within the circles correspond to the atomic oxidation state changes of the Nitrogen ion $\mathtt{N}^{+}$ and the Oxygen ion $\mathtt{O}^{-}$ respectively.

Figure 3.3 shows a different odd-cycled arrangement of the ITS, that can occur due the presence of a single charged atom. The unshared electron pair of the Sulfur ($\mathtt{S}^{-2}$) ion contributes to the formation of two adjacent bonds resulting in an odd ring. This case of bond and atoms valence changes within the ITS requires a dedicated treatment i.e.

[3]Terminologies: non-bonding electron pair, unshared electron pair, and delocalized electron pair are similar

dedicated CSP formulation (presented in 3.4.2.2) which is different from the CSP with two oppositely charged atoms.

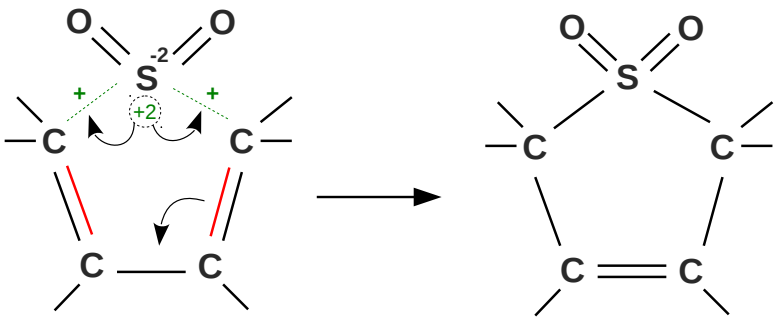

FIGURE 3.3: Sulfur Dioxide Cycloaddition, adapted from [4]. The change of delocalized electron pair into two bonds. Red bonds are broken, green dotted bonds are formed. The number within the circle corresponds to the charge change of the Sulfur ion S^{-2}.

Chemical reactions with charged atoms in the ITS feature mostly an odd-numbered cycle[4], since the unshared electron forms bond pair in product. So they usually involve 3, 5 or 7 atoms in their ITS ring. Meisenheimer rearrangement fig. 3.2 and Sulfur dioxide cycloaddition fig. 3.3 exemplify an elementary ambivalent chemical reaction with 5-cycled ITS.

In case of an odd ITS ring, it is not possible to find a simple circular ITS using the current CSP, since there are bonds which do not alternate. An extra rule for charge-bond changing of redox atoms between educts and products has to be introduced, too. Furthermore we have to take into consideration different ITS layouts dictated by charged atoms. These reasons impose an extension of our constraint-based model to enable the adaptation of these factors, presented in the following section.

[4]Ambivalent reaction can also have even-numbered cycles, but it is less frequent than odd-numbered cycles.

3.4.2 Odd CSP Formulation

An ambivalent CSP is an extension of the elementary homovalent CSP with respect to the required changes mentioned above and it is formulated for different odd ring sizes $k \in \{3, 5, 7\}$. The main difference when formulating an odd CSP is that homovalence is not enforced for all participating atoms. Besides, the ambivalent CSP incorporates an extra constraint regarding charged atoms. Here, the atom **charge change** constraint is responsible for tracing the atoms whose atomic oxidation state changes during the reaction and it says: $(\mathcal{I}_{X_i^I,X_i^I} - \mathcal{O}_{X_i^O,X_i^O}) = 1$. In a broad sense, the diagonal entries of the adjacency matrices $\mathcal{I}, \mathcal{O}$ representing non-bonding electrons are not constant in case of charged atom and radicals. Note that the homovalence constraint 3.2.1 is a special case of the charge change constraint, in which the number of non-bonding electrons in the atoms does not change i.e. the change in the charge is equal to null, $(\mathcal{I}_{X_i^I,X_i^I} - \mathcal{O}_{X_i^O,X_i^O}) = 0$.

It is important to know that the elementary odd CSP does not enforce the introduced order constraints presented for the elementary homovalent CSP, since the odd ITS shows no rotation symmetries. The charge change constraint still poses a very strong constraint sufficient to ensure the good performance.

We formulate here separate odd CSPs with respect to odd ITS ring layouts. The existence of one or multiple charged atoms and their positions in the chemical molecule cause different arrangements of the ITS ring and thus different layouts to be considered. In the following we build for each ITS layout case a suitable odd CSP and show the differences in posting the constraints on ambivalent atoms.

3.4.2.1 Layout-1: Two Oppositely Charged Atoms

This layout features chemical reactions that include two connected, oppositely charged atoms as illustrated in the reaction from the fig. 3.2. In the course of such a redox reaction, the negative and positive ions in educts turn into neutral atoms in the products. The figure 3.4 sketches this odd arrangement of the ITS. Considering the layout below, the positive ion has the charge change (-1) due to the lack of an electron, so it receives an electron from its adjacent bond resulting in bond breakage. Yet the charge change

(+1) corresponds to the negative ion that denotes its extra electron, which forms a new covalent bond with a neighbour atom.

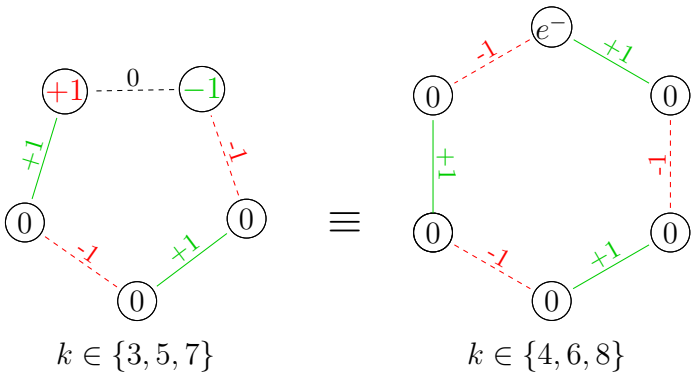

FIGURE 3.4: ITS layout-1: two oppositely charged atoms. The number within the nodes corresponds to atomic oxidation state changes, red dotted bonds are broken, green bonds are formed, the black dashed bond is preserved. On the right, an equivalent layout for the next larger even cycle with a pseudo-node labelled (e^-).

For the determination of this kind of odd-cycled ITS[5], we post charge change constraints on both charged atoms and homovalence constraints on the remaining atoms, since they have no charge changes. For this case the alternating cycle constraint holds for all ITS bond pairs except the bond connecting the oppositely charged atoms. Such bonds are preserved (do not change) and treated as pseudo-alternating bonds. This holds for instance for the bond connecting $\mathtt{N}^+$ and $\mathtt{O}^-$ in the Meisenheimer rearrangement 3.2, which is reflected in layout figure 3.4 (left) as black dashed edge labelled with 0.

The algorithm 2 sketches the CSP setup for this odd ITS case. We extend the original constraint-based model and perform moderate changes in the formulation of some constraints according to the requirements dictated by the underlying ITS layout.

3.4.2.2 Layout-2: Single Ambivalent Atom

The observed odd CSP here considers the presence of just one ambivalent atom. This enforces the ITS cycle to take different odd rearrangements from the described above. Depending on the charge of the single atom, the underlying ITS can form two different

[5] As we will see in the next chapter 4, this layout is encoded using the generic format for ring size ($k = 5$) as `[+1]+[0]-[0]+[0]-[-1]=`.

Algorithm 2 Identification of odd ITS with two oppositely charged atoms

Require: eduAtoms, proAtoms are arrays of size $k \in \{3, 5, 7\}$
Ensure: odd ITS of size k

▷ Bijective mapping between educt and product
distinct(eduAtoms)
distinct(proAtoms)

▷ Considering homovalent atoms
for $i = 2 \rightarrow k - 1$ **do**
  preserveLabel(eduAtoms[i],proAtoms[i])
  homovalent(eduAtoms[i],proAtoms[i]) = **chargeChange**(eduAtoms[i],proAtoms[i],0)
  edgeDegree(eduAtoms[i],proAtoms[i],1) ▷ Loss or gain or bonds is bounded by 1
end for

▷ Charge change for ambivalent atoms
chargeChange(eduAtoms[1],proAtoms[1],-1)
chargeChange(eduAtoms[k],proAtoms[k],+1)

▷ Label preservation and edge degree for ambivalent atoms
preserveLabel(eduAtoms[1],proAtoms[1])
preserveLabel(eduAtoms[k],proAtoms[k])
edgeDegree(eduAtoms[1],proAtoms[1],1)
edgeDegree(eduAtoms[k],proAtoms[k],1)

▷ Ensure alternating cycle structure of the ITS in the mapping without closing the last ring pair
for $i = 1 \rightarrow k - 1$ **do**
  if $(i \; mod \; 2 = 0)$ **then** ▷ Bond formation in case of even indices
    alternateCycle(eduAtoms[i],eduAtoms[i+1],proAtoms[i],proAtoms[i+1],1)
  else ▷ Bond breakage in case of odd indices
    alternateCycle(eduAtoms[i],eduAtoms[i+1],proAtoms[i],proAtoms[i+1],-1)
  end if
end for

▷ Ensure ring closure shows no bond valence change (non-changing bond)
alternateCycle(eduAtoms[k],eduAtoms[1],proAtoms[k],proAtoms[1],0)

layouts shown in the figure 3.5. When the charge change corresponds to (-2), the ambivalent (positive) atom obtains two electrons from the breakage of two adjacent bonds to compensate its missing electrons (the right case in the figure). Accordingly, in case of charge change of (+2), the negatively charged atom denotes its additional electrons to form two new adjacent edges (left part of the figure. Also see $\mathtt{S}^{-2}$ in fig. 3.3). Note the absence of preserved bonds in this layout when compared to the previously mentioned layout, in which two oppositely charged atoms are connected via a non-changing bond. The one-ambivalent atom CSP differentiates in its formulation from the two-charged atoms CSP in three aspects:

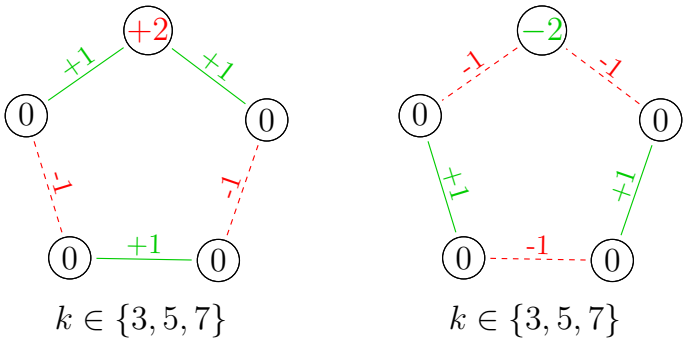

FIGURE 3.5: ITS layout-2: single charged atom. The number within the nodes corresponds to atomic oxidation state changes, red dotted bonds are broken, green bonds are formed.

- The charge change constraint is posted only once, since there is only a single charged atom.
- The edge degree by ± 1 constraint does not hold any more for this ambivalent atom. The charged atom loses or gains two bonds at once, thus the edge degree constraint posted on the ambivalent atom is bounded by two.
- The bonds adjacent to the ambivalent atom violate the alternating cycle constraint, since both are either formed (+1) or broken (-1).

The implementation of these aspects is outlined in the algorithm 3.

So far in this thesis, we presented different CSP formulations regarding reaction kind, homovalent or ambivalent and optimization option. In case of ambivalent reactions, a number of separate CSPs are formulated depending on the observed ITS cycle. From practical point of view, it is desirable to integrate all those cases with corresponding optimizations in one generic CSP. This would avoid the formulation of several CSPs, especially for the odd ITS cycles. A generic prototype would allow to skip the special treatment needed for charged atoms in terms of charge change and edge degree constraints. For this purpose, we introduce in the next chapter a generic atom mapping framework that employs a generic ITS encoding suitable for all already used ITS arrangements and flexible to incorporate new ones.

Algorithm 3 Identification of odd ITS with single atom with charge change of -2

Require: eduAtoms, proAtoms are arrays of size $k \in \{3, 5, 7\}$
Ensure: odd ITS of size k

▷ Bijective mapping between educt and product
distinct(eduAtoms)
distinct(proAtoms)

▷ Considering homovalent atoms
for $i = 2 \rightarrow k$ **do**
  preserveLabel(eduAtoms[i],proAtoms[i])
  homovalent(eduAtoms[i],proAtoms[i])
  edgeDegree(eduAtoms[i],proAtoms[i],1) ▷ Loss or gain or bonds is bounded by 1
end for

▷ Charge change for the single ambivalent atom
chargeChange(eduAtoms[1],proAtoms[1],-2)

▷ Label preservation and edge degree (bounded by 2) for the ambivalent atom
preserveLabel(eduAtoms[1], proAtoms[1])
edgeDegree(eduAtoms[1], proAtoms[1], 2)

▷ alternating cycle structure of odd the ITS in the mapping
for $i = 1 \rightarrow k - 1$ **do**
  if $(i \, mod \, 2 = 0)$ **then** ▷ Bond formation in case of even indices
    alternateCycle(eduAtoms[i],eduAtoms[i+1],proAtoms[i],proAtoms[i+1],+1)
  else ▷ Bond breakage in case of odd indices
    alternateCycle(eduAtoms[i],eduAtoms[i+1],proAtoms[i],proAtoms[i+1],-1)
  end if
end for

▷ Ensure ring closure shows bond breaking
alternateCycle(eduAtoms[k], eduAtoms[1], proAtoms[k], proAtoms[1], -1)

Chapter 4

Generic Atom Mapping Framework

This part explains in detail the generic atom mapping framework which utilizes a generic ITS-based CSP formulation. The generic CSP unifies all mentioned formulations, since it operates directly on a given ITS graph layout encoding able to describe all previously shown ITS layouts. The generic framework introduces an advanced method for the exclusion of symmetries. We also outline the implementation details of the framework.

4.1 Generic ITS Encoding

So far we have described several CSP variants for handling different layouts of the imaginary transition state. The elementary homovalent CSP 3.2.1 is formulated when the ITS is an even-numbered cycle. Separate optimized CSPs 3.3 are formulated based on the elementary homovalent CSP, since they extend the basic version by employing additional constraints. For odd ring sizes we have seen the elementary ambivalent CSP 3.4.2, which also has different variants implied by the odd-cycled layouts. In the previous part only two odd cases are presented, namely "Two Oppositely Charged Atoms" 3.4.2.1 and "Single Ambivalent Atom" 3.4.2.2. Consequently new observed layouts of the ITS impose the development of new CSPs i.e. according modifications in the formulation of the constraints have to be performed. Due to the variety of chemical reactions and

correspondingly their ITS layouts, it is not desirable from the practical point of view to have many separate CSP implementations. To capture almost all possible layouts of the ITS in a single generic CSP prototype and to avoid the formulation of lots of CSP variants, we introduce a generic encoding of the ITS cycle.

It is possible to represent elementary ITS cycles through a generic string encoding. The ITS-participated atoms are represented by the change in their charge. For instance, in homovalent reactions the non-bonding electrons remain unchangeable, so all atoms that form the ITS are described as [0]. Alternatively, charged atoms in the ambivalent reaction are merely expressed by change in the oxidation state (delocalized electron pairs). Thus the encoding of negative ions (e.g. $\mathtt{S}^{-2} \rightarrow \mathtt{S}$ in the fig. 3.3) which are willing to denote unshared electrons to form covalent bonds is `[+1], [+2],...,` etc. since their charge is increased. However positively charged atoms, that lack in electrons filled by the breaking of an adjacent bond, show a negative charge change of `[-1], [-2],...,` etc. such as $\mathtt{N}^{+} \rightarrow \mathtt{N}$ in the fig. 3.2. For more information on the atomic oxidation state see previous sections 3.4.2.1, 3.4.2.2.

We still have to include in the ITS string the change in the bond order (edge valence) between the adjacent atoms which form the ITS sequence. For this purpose we encode the change in the valence electrons of the right edge and here there are three cases $\{+, -, =\}$ representing bond formation, bond breaking, and bond conservation, respectively.

Given that, we can encode a string notation which is sufficiently able to express arbitrary ITS formats. Each neutral/charged atom participated in the ITS is described by the corresponding charge change as an integer number between brackets, such that `[`$\pm n$`]`. Every bond connecting the adjacent atoms within the ITS sequence is expressed by one of the three operators $\{+, -, =\}$ regarding the bond change (formation, breaking, or no change). These operators indicate always the change in the right bond order of the underlying ITS atom. The ring closure is also encoded using one of these operators which is placed at the end of the string. For example, the six-membered ITS of the Diels Alder 1.3 homovalent reaction is written as `[0]+[0]-[0]+[0]-[0]+[0]-`. The alternating cycle of this ITS is illustrated using bond formation (`+`) and bond breakage (`-`) of the right ITS edge. Considering the ambivalent Meisenheimer rearrangement 3.2, it exhibits a five-membered ITS cycle with charge change of `[+1]` in case of $\mathtt{O}^{-}$ (electron participates in a bond) and the charge change of `[-1]` at nitrogen $\mathtt{N}^{+}$ (compensation of

the missed electron through bond breakage). Additionally the bond between $\texttt{O}^-$ and $\texttt{N}^+$ in the ITS ring is conserved. So the ITS of the Meisenheimer reaction from the figure 3.2 is encoded by `[+1]+[0]-[0]+[0]-[-1]=`. This notation is a variant of the notation introduced by Hendrickson [4].

Using this notation we can simply define any bond and atom valence changes within the ITS as a string. The string syntax is then parsed into an ITS graph. Furthermore, the adjacency information of the constructed ITS graph is encoded in a matrix C, whose diagonal entries correspond to charge changes in the associated encoding, whereas the remaining entries contain bond changes of the ITS string. Based on the ITS graph and its matrix C, the generic CSP is formulated and solved. To support a new layout, it is only required to encode using the syntax above. The following table 4.1 lists ITS layouts (for different ITS ring sizes), which are currently supported by the atom mapping framework.

k	ITS-Encoding

3	`[+2]+[0]-[0]+`
	`[-2]-[0]+[0]-`
	`[+1]+[0]-[-1]=`

4	`[0]+[0]-[0]+[0]-`

5	`[+1]+[0]-[0]+[0]-[-1]=`
	`[+2]+[0]-[0]+[0]-[0]+`
	`[-2]-[0]+[0]-[0]+[0]-`

6	`[0]+[0]-[0]+[0]-[0]+[0]-`

7	`[+1]+[0]-[0]+[0]-[0]+[0]-[-1]=`
	`[+2]+[0]-[0]+[0]-[0]+[0]-[0]+`
	`[-2]-[0]+[0]-[0]+[0]-[0]+[0]-`

8	`[0]+[0]-[0]+[0]-[0]+[0]-[0]+[0]-`

TABLE 4.1: List of ITS layouts currently supported by the generic atom mapping framework

The figure 4.1 illustrates different ITS layouts for the table 4.1 supported by the generic approach.

4.2 ITS Selection

From now on, we no longer need to differentiate between different ITS ring layouts. In order to determine reaction mappings, we first need to determine the suitable ITS from

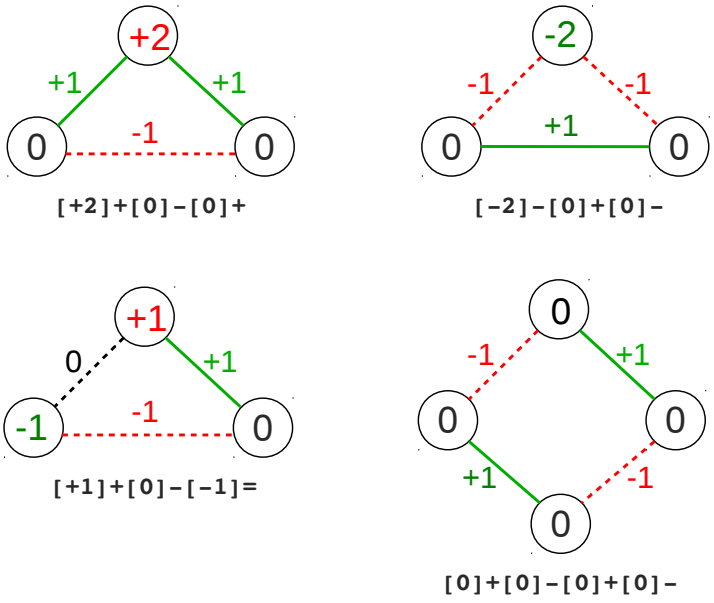

FIGURE 4.1: Currently available ITS layouts (smallest variant for each type) with the associated string encoding. The number within the nodes corresponds to charge changes, red dotted bonds are broken, green bonds are formed, the black dashed bond is preserved.

the list of available layouts (see table 4.1) associated with the observed reaction. After the selection of an appropriate ITS, the generic CSP can be formulated.

We mentioned in the section 1.3 that we have to deal with three variants of the chemical reaction mapping problem. *Decision*, *Optimization*, and *Enumeration* of the atom mapping are covered through the selection procedure of the generic framework as following:

1. **Decision Problem:** Whether or not there is an atom mapping with associated cyclic ITS of length k. The generic framework merely scans the list of ITS layouts using the given size k and detects the appropriate layout for that reaction.

2. **Optimization Problem:** Find an atom mapping associated with minimal length k of an ITS. In this case the ITS size is not given, so the framework scans the ITS list in an increasing order and finds the smallest ITS associated with the reaction that yields mappings.

3. **Enumeration Problem:** Find all mappings associated with an ITS of length k. Of course the framework gives the option to enumerate atom maps for all valid ITS layouts of a reaction.

4.3 Generic CSP Formulation

The next step after the selection of an appropriate ITS is to formulate and solve the generic constraint satisfaction problem. Given the educt molecule graph $I = (V_I, E_I)$, the product molecule graph $O = (V_O, E_O)$ and the ITS for k atoms with adjacency matrix C, we define the generic CSP for the given k ITS atoms. The matrix C holds all ITS changes encoded in the string notation, i.e. $C_{i,j} = bondchange$ and $C_{i,i} = chargechange$ for $1 \leq i \leq j \leq k$, where $bondchange \in \mathbb{N}$ and $chargechage \in \mathbb{N}$. We encode k ITS variables in the educts $\{X_1^I, \ldots, X_k^I\}$ and corresponding k variables in the products $\{X_1^O, \ldots, X_k^O\}$ with the domains $D_i^I = V_I$ and $D_i^O = V_O$ representing the nodes in the educt (V_I, E_I) and product (V_O, E_O) graphs respectively. In order to identify the ITS subgraph common to educt and product molecule graphs, the following constraints must be satisfied:

1. **Bijective Mapping:** All variables must be assigned distinct values in order to ensure bijective mapping, i.e. $\forall i \neq j : X_i^I \neq X_j^I$ and $\forall i \neq j : X_i^O \neq X_j^O$.

2. **Label Preservation:** An atom label is given as $l(x)$ for $x \in V_I \cup V_O$. The corresponding atom labels between educts and products must be equal $l(X_i^I) = l(X_i^O)$, i.e. we have to enforce $\forall e \in D_i^I : \exists p \in D_i^O : l(e) = l(p)$ as well as $\forall p \in D_i^O : \exists e \in D_i^I : l(p) = l(e)$.

3. **Edge Degree:** This constraint requires a moderate change in its formulation. The loss or gain of edges is no longer generally bounded by one. The generic ITS encoding includes charged atoms which can gain or lose more than one edge depending on their oxidation state change (see section "Layout-2" in the odd CSP 3.4.2.2). Hence the edge degree is enforced through the ITS encoding, such that $|degree(X_i^I) - degree(X_i^O)| \leq max(1, C_{i,i})$, where $C_{i,i}$ gives the charge change of ITS node i.

4. **Charge Change:** The charge change constraint shown in "Odd CSP Formulation" 3.4.2 can be used to combine here the functionality of the homovalence and the ambivalence. Case distinction between homovalent and ambivalent atoms is no longer required, since it is encoded in the ITS string. In this sense, atom mapping must preserve the change in the atomic oxidation state at the corresponding position i of the ITS i.e. diagonal entries in the adjacency matrices $\mathcal{I}$, $\mathcal{O}$ must satisfy $\mathcal{I}_{X_i^I,X_i^I} - \mathcal{O}_{X_i^O,X_i^O} = C_{i,i}$.

5. **Ring Bonding:** Ring bonding replaces the alternating cycle condition of the basic formulation. The ITS ring in the generic formulation does not exhibit only an alternating cycle structure due to the involvement of odd arrangements. In this case ring pair indices (i, j) of the ITS sequence 1-2-..-k-1 are not distinguished in terms of even/odd indices for bond formation or breakage. They only have to conform to the bond change at the respective positions of the ITS encoding, which means $\mathcal{O}_{X_i^O,X_j^O} - \mathcal{I}_{X_i^I,X_j^I} = C_{i,j}$. Note that the bond change operators $\{+, -, =\}$ are encoded as integer values $\{1, -1, 0\}$ in the matrix C.

6. **Coverage of Connected Components:** A chemically correct mapping should cover all molecules in the educts and in the products accordingly and avoid that only atoms of some molecules are present in the ITS cycle. In other words, we have to ensure that at least one atom of each educt and product molecule is participated in the ITS ring. Given V_I we denote with V_I^1 the node set of the first educt molecule, thus it holds $V_I = \biguplus_x V_I^x$ and $\forall x \neq y : V_I^x \cap V_I^y = \phi$. V_O is defined accordingly. We therefore have to ensure $\forall x : \exists i : D_i^I \cap V_I^x \neq \phi$ and $D_i^O \cap V_O^x \neq \phi$. The example in 4.5.2 shows mapping results before and after using the underlying constraint.

7. **ITS Educt Symmetry:** Given an ITS, we can find the needed order constraints to break rotation symmetries. These constraints are posted on ITS educt variables. Details how the constraints are found is discussed in 4.4.2.

8. **Minimal Edge Valence of ITS Atoms:** This constraint is previously mentioned in 3.3.1 and is used as an improvement to speed up the ring bonding constraint. For all bond formation pairs, a minimal product bond valence of one is enforced i.e. $\mathcal{O}_{X_i^O,X_j^O} \geq 1$ and for all bond breaking pairs, a minimal educt bond valence of one is enforced i.e. $\mathcal{I}_{X_i^I,X_j^I} \geq 1$.

9. **Local ITS Neighbourhoods:** This constraint is taken from the extended CSP version 3.3.2. It aims at raising the efficiency of the generic CSP through the precomputation of a lower bound on the atom types that are part of the ITS and their neighbourhood.

10. **ITS Atoms Count:** This constraint is based on the local ITS neighbourhoods. Once the lower bound of the ITS-participated atoms is fixed, we constrain the educt variables to conform to the precomputed number of occurrences of the ITS atoms. Given a precomputed atom type (label) $l(x)$ for $x \in V_I \cup V_O$, we denote n_x the number of the appearances of this atom label in the ITS. We have to enforce the occurrence of each identified atom label for educts, so $|\{X_i^I | l(X_i^I) = l(x)\}| \geq n_x$. This is automatically propagated on product variables X^O via the atom label preservation constraints.

The importance of the connected component constraint is shown when testing the reaction R5 from the table 5.1 (chapter "Tests and Evaluation"). Before ensuring the coverage of all connected components, the generic CSP yields 120 overall mappings for this reaction. Constraining the mapping to cover all molecules contributes to the reduction in the number of candidates to 4. Many invalid mappings are this way excluded from the final result.

4.4 Symmetry Elimination

We mentioned the problem of equivalent mappings in 3.3 while enumerating all atom mappings. Due to the exchangeability of atoms, the constraint-based approach locates several mappings, which corresponds to the same reaction mechanism [23]. Considering for example the carbon dioxide molecule `O=C=O`, an atom mapping can result in the permutations $\big((\texttt{O}:2),(\texttt{C}:1),(\texttt{O}:3)\big)$ or $\big((\texttt{O}:3),(\texttt{C}:1),(\texttt{O}:2)\big)$. These assignments (2, 1, 3) and (3, 1, 2) do not yield distinct reaction mechanisms, so we have to omit such atom maps.

During this work we had to deal with three cases of symmetry: hydrogen symmetries, ITS symmetries, and the general case of educt/product symmetries. A lot of symmetries arise due to reshuffling of hydrogen atoms within the ITS. The hydrogen symmetries can be

already excluded during the processing of a chemical reaction and before the formulation of the generic CSP as we will see in 4.4.1. To avoid rotation symmetric assignments of the ITS, one has to define ordering conditions on the ITS atoms. However these two aspects alone are not sufficient to avoid the permutation of educt/product mappings. Therefore the atom mapping framework features a method to exclude symmetries dynamically during the search.

For a given graph $G = (V, E)$ a symmetry s is an injective function $s : V \to V$ that maps each node of the graph onto its symmetric equivalent due to rotation or reshuffling [35]. We introduce S the set of all such symmetries $s \in S$ in V, so the identity relation $s'(v \in V) = v$ is a symmetry of S. The set of all symmetries S can be used to convert symmetric atom mappings to unique mappings as discussed in the following sections. This is done via a tabularization of symmetric mappings serving as lookup tables of the symmetry functions s. Given a index order on V, then the symmetries $(V_1, V_2, ..., V_n) = (r(V_1), r(V_2), ..., r(V_n))$ provide the index shuffle table. For instance, given an ITS with $k = 4$, the shuffle tables regarding the ordered ITS assignment (1,2,3,4) are (2,1,4,3), (3,4,1,2), and (4,3,2,1).

4.4.1 Exclusion of Hydrogen Symmetries

Investigating the given educt and product graphs, it is possible to exclude a large set of symmetric solutions that arise due to an exchange of hydrogens. The chemical specification of the hydrogen allows it to form at most one single bond to other atoms. Thus, if a hydrogen participates in the ITS, its adjacent atom will do as well. Most adjacent atoms are non-hydrogens, like carbon atoms, can have multiple adjacent hydrogens. Since there is exactly one bond breaking and formation for each ITS atom, only one such adjacent hydrogen will be part of the ITS. This results in a vast combinatorial symmetry due to the replacement of the hydrogen through its "sibling" hydrogen atoms. An example is given in figure 4.2.

To exclude this type of symmetry, we define for each non-hydrogen one "master" hydrogen and remove all other "sibling" hydrogens from the domains, both for educt and product variables X^I and X^O, resp., before starting the CSP solving. Note that the assignment of the "master" hydrogen does not violate the adjacency information.

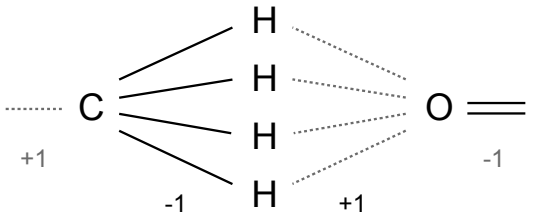

FIGURE 4.2: Symmetries resulting from interchangeable hydrogens. The figure presents three successive atom assignments within an ITS mapping. Bonds present in I are given in black, bonds to be formed to derive O are dotted and gray. The ITS describes the loss of an hydrogen for the carbon (bond order decrease) and the bond formation between the decoupled hydrogen with the oxygen next in the ITS. It becomes clear that all 4 hydrogens are not distinguishable, which results in 4 possible symmetric ITS mappings. Source: Atom Mapping with Constraint Programming [5].

4.4.2 Exclusion of ITS Symmetries

As mentioned in the basic CSP 3.2.1 and in the generic CSP 4.3 formulations, we have to post specific constraints to avoid the problem of symmetric ITS matches on itself. This kind of isomorphism is nothing else but rotation or reflection assignments of the ITS graph. To overcome the ITS symmetries, we enforce ITS-specific order constraints to be posted on ITS educt variables. We will demonstrate the derivation of the needed order checks using a homovalent ITS of a size $k = 4$. The ITS assignment (1,2,3,4) can be shuffled to produce the following symmetries: s^1 =(1,2,3,4), s^2 =(2,1,4,3), s^3 =(3,4,1,2), and s^4 =(4,3,2,1) as demonstrated in the figure 4.3.

Given an ITS graph C, we denote S^C the set of all ITS symmetries in the educt mapping such that $S^C = \{s^1, s^2, s^3, s^4\}$. In order to break ITS symmetries, we have to enforce that only one symmetry $s^* \in S^C$ is found and all other $s^i \neq s^* \in S^C$ are not enumerated. This is done via a set of binary order constrains $C_{order} = \{(X_i^I, X_j^I) | 1 \leq i < j \leq k\}$, which are compatible with s^* but violated by all $s^i \neq s^*$.

Before posting binary order constraints on ITS educt variables, we need to generate the required order checks used by the underlying constraints to counter symmetric ITS candidates. The order checks act as a filter, so that only the identity assignment of the ITS s^1 =(1,2,3,4) is captured and all other self matches $s^i \neq s^1$ are omitted. For this purpose, we create an order check list of the assignments in S^C by testing each symmetry $s^i \in S^C$ whether every pair of elements (s_j^i, s_{j+1}^i) conforms to the lexicographic order ($\leq$). If a pair violates the required ordering, its assignment has to be deleted from S^C.

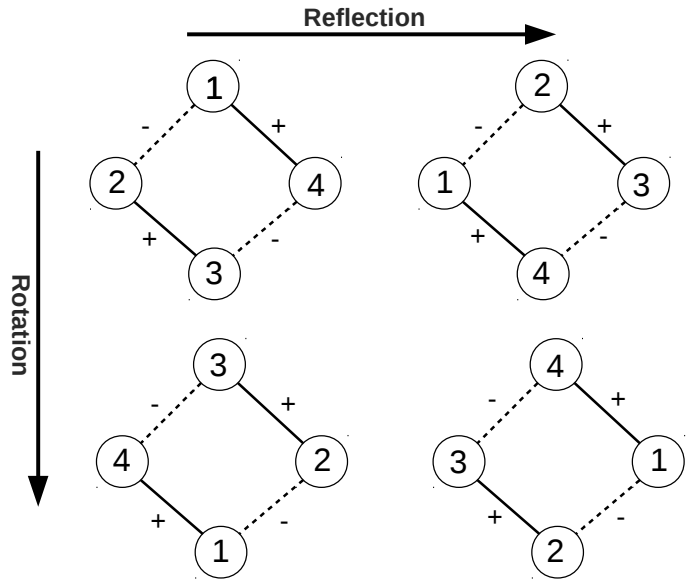

FIGURE 4.3: Symmetric assignments of an ITS with $k = 4$.

For the example above, we notice that the assignments s^2, s^3 and s^4 do not meet the enforced ordering and thus they correspond to ITS symmetries. Consequently it remains only the ITS identity assignment $s^* = s^1 = (1,2,3,4)$ as unique ITS mapping of the educt vertices and all other ITS rotations/reflections are broken.

The ITS order checks are generated, once the ITS is selected i.e. before the CSP formulation. However, constraining the ITS educt variables to conform to the order checks is performed afterwards during the CSP. The generation of the order checks is done by the method getGraphAutomorphism() of the Graph Grammar Library (GGL) [36].

4.4.3 Exclusion of Educt/Product Symmetries

Here we eliminate symmetries during the search for ITS mapping. The main idea of this method is to generate for each CSP solution all according symmetric assignments and save them for successive filtering of the following solution candidates. We denote the educt symmetry as $s_I : V_I \rightarrow V_I$ and the product symmetry as $s_O : V_O \rightarrow V_O$ with the corresponding sets S^I, S^O that store all symmetric assignments of the educt and the product respectively. For each educt/product solution, all according swapping

assignments of the underlying ITS are generated. We observe as example the homovalent reaction R2 from the table 5.1 (chapter "Tests and Evaluation") with domains $D^I = \{1..6\}, D^O = \{7..12\}$ as depicted in the figure 4.4 below. The associated shuffle symmetries of the educt are $S^I = \{(1,2,3,4,5,6),(4,5,6,1,2,3)\}$, whereas the product has only one symmetric assignment $S^O = \{(7,8,9,10,11,12)\}$, and the ITS symmetries correspond to $S^C = \{(1,2,3,4),(2,1,4,3),(3,4,1,2),(4,3,2,1)\}$.

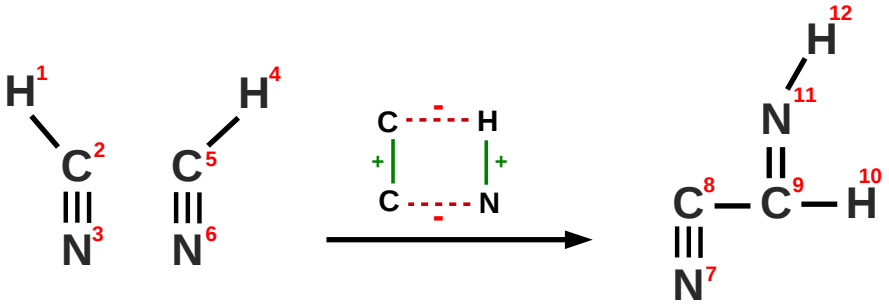

FIGURE 4.4: The homovalent reaction R2 from the table 5.1 with the underlying ITS. Broken bonds are dotted red, formed bonds are green.

One possible ITS mapping is $X^I = (5,4,3,2)$ in the educt and $X^O = (8,12,11,9)$ in the product. Given S^I, S^O, S^C we can identify all symmetric assignments by generating all ITS symmetries via S^C that apply S^I and S^O i.e. ITS matching positions of the assignments of X^I and X^O respectively. This results in the following permutations:

$$
\begin{aligned}
&S^C(X^I) = \{(5,4,3,2),(2,3,4,5),(3,2,5,4),(4,5,3,2)\}\\
&S^I(S^C(X^I)) = \{(5,4,3,2),(2,3,4,5),(3,2,5,4),(4,5,2,3),(2,1,6,5),(5,6,1,2),\\
&\qquad\qquad (6,5,2,1),(1,2,5,6)\}\\
&S^O(S^C(X^O)) = \{(8,12,11,9),(9,11,12,8),(11,9,8,12),(12,8,9,11)\}
\end{aligned}
$$

These sets represent all possible symmetries. Now we add additional constraints to the CSP that forbid the assignments $S^I(S^C(X^I) \setminus \{X^I\}$ and $S^O(S^C(X^O)) \setminus \{X^O\}$. In other words, the constructed symmetry sets $S^I(S^C(X^I)$ and $S^O(S^C(X^O))$ serve as a lookup to filter ITS solution candidates for educt X^I and X^O product variables, that show symmetries.

To this end, we split variables search DFS(X) in hierarchical search i.e we look for educt assignments DFS(X^I) and afterwards for product assignments DFS(X^O). Thus for each

educt assignment X^I, we start searching for the corresponding product assignments DFS(X^O) and initialize thereby a set of symmetric product solutions to omit $A^O = \phi$. Each X^O assignment is examined for symmetries regarding the membership in A^O i.e. if $X^O \in A^O$, then the found assignment X^O is symmetric and has to be ignored. In case $X^O \notin A^O$, all possible ITS symmetries that apply the current product assignment X^O are generated and added to the omission set such that $A^O = A^O \cup S^O(S^C(X^O)) \setminus \{X^O\}$. In this sense, each consequent product solution candidates is filtered using the set A^O, which avoids the production of symmetric product assignments. Note that the omission set A^O has to be reinitialized for each new educt assignment. The search for educt variables DFS(X^I) is performed in the same way with the according omission set A^I, however A^I is not reseted during the search. As a result of this hierarchical search, we get no symmetric solutions for both X^I and X^O and thus the enumeration of symmetry-free ITS candidates is guaranteed.

4.4.4 Exclusion of Symmetries of Overall Atom Mapping

The problem of hydrogen reshuffling and symmetries is faced again when deriving the overall atom mapping via VF-2 graph matching. In order to exclude symmetric overall atom mappings, we produce here intermediate "compressed" educt/product graphs, where all adjacent hydrogens which are not part of the ITS are collapsed into the atom labels of their adjacent non-hydrogens atoms. For each overall atom mapping, we generate then all possible symmetric assignment of the current mapping and report the first/smallest symmetry, i.e. we associate the smallest symmetry with the unique reaction mechanism.

4.5 Implementation Details

4.5.1 Preprocessing of Chemical Reactions

Our C++ implementation of the approach uses reaction SMILES [6] notation to represent chemical reactions. In order to define reaction SMILES, we firstly give an insight to the SMILES notation. SMILES (Simplified Molecular Input Line Entry System)

is a computerized chemical notation used to describe molecular structure. Molecular compounds are simply expressed in SMILES as a linear string. SMILES is developed to be an easy-readable and machine-independent format. There are several rules and algorithms governing the unique generation of SMILES. The rules of SMILES concern atoms, bonds, branches, cycles, and aromaticity specifications of molecules. Molecular structures encoded in SMILES can be decoded into a graph, which offers an important simplification of chemical structure similar to chemists' view of molecules.

Just as a SMILES represents a molecule, a reaction SMILES represents the molecules in a chemical reaction [1]. Reaction SMILES consist of educt and product molecules separated by ">>". In case there are several molecules in educts or products, they are combined using ".". Hydrogen atoms are optional in the SMILES syntax and they are normally omitted to make the SMILES more compact and readable. The following example depicts a reaction SMILES for the chemical reaction R5 taken from the table 5.1 in the chapter "Tests and Evaluation" 5:

```
O.Cl.CC(=O)OCC>>Cl.OCC.CC(=O)O
```

This reaction is depicted in form of molecule graph in the figure 4.5.

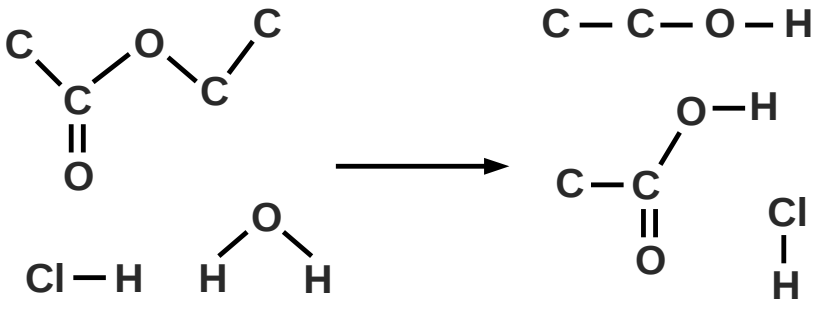

FIGURE 4.5: Molecule graph of the reaction R5.

Molecule parsing, writing, and graph representation use the chemistry module of the Graph Grammar Library (GGL) [36]. Given the educt/product graphs, we perform various precomputations. We represent explicitly the hydrogen atoms within the CSP formulation, since most elementary reactions involve a replacement of at least one hydrogen in the ITS cycle. The compact string encoding of molecules in SMILES format

does not explicitly represent hydrogens, so we use the hydrogen filling procedure of the GGL to complete educt and product molecule input. The hydrogens adjacent to each non-hydrogen atom are then replaced through a "master" hydrogen. Furthermore, we determine the local neighbourhoods within the educts and the products discussed in 3.3.2, that are part of the ITS. This is followed by the derivation of the minimal bound of atoms participated in the ITS. Besides, we identify the order checks mentioned in 4.4.2, which are required to exclude the rotational assignments of the ITS. The preprocessing of the chemical reaction is followed by the selection of the appropriate ITS layout for the underlying reaction and the formulation of the generic CSP.

4.5.2 Generic CSP Implementation and DFS-Search

The CSP formulation and solving is done within the Gecode framework [29] using integer encodings of the atom indices. The generic CSP uses standard binary propagators and distinct n-ary propagators provided by the Gecode library to implement the combinatorial constraints. Dedicated binary constraints propagating on unassigned domains have been implemented for preservation of atom label, edge degree, minimal edge valence, charge change and homovalence.

The ring bonding is implemented by a sequence of 4-ary constraints propagating on the bond order change of the ITS edges. The ITS local neighbourhood enhancement to be enforced in the extended CSP and in the generic framework is implemented by a dedicated n-ary propagator over all variables, which propagates on full assignments only. Similarly an n-ary propagator is used for connected component coverage constraint, which propagates on unassigned domains.

The domains restriction for the variables $\{X_1^I, \ldots, X_k^I\}$ and $\{X_1^O, \ldots, X_k^O\}$ during propagation is done via "support sets". For all constraints except "Atom Label Preservation", we collect domain values which satisfy the constraints in corresponding sets. Then we replace the domains of the according variables through the contents of the support sets using the Gecode method "narrow_r". In case of the "Atom Label Preservation", domain pruning is done in a reverse manner. The domain values which do not fulfil the constraints are collected in "delete sets". After that we apply the Gecode function "minus_r" to subtract the delete sets from the variable domains.

We are using a Depth-First-Search (DFS) where the branching strategy chooses first variables with minimal domain size and first assigns non-hydrogen node before hydrogen nodes are considered. The latter increases the performance to find the first solution, since most reaction mechanism are constructed of at least 50% non-hydrogen atoms. The ITS in the following figure 4.6 contains two hydrogens out of six atoms. Once a non-hydrogen is selected, propagation will ensure that adjacent hydrogens are considered for the neighboured variables within the ITS ring encoding.

4.5.3 VF-2 Graph Matching and Generation of Mapped Reaction SMILES

Each ITS solution candidate given by the generic CSP is followed by a graph matching procedure to derive the overall mapping. As mentioned before, this is done via the VF2-algorithm [31] that is implemented in the subgraph matching module of the GGL. Furthermore, the atom mapping framework produces during the final graph matching the compressed educt/product graphs to eliminate symmetric overall mappings mentioned in 4.4.4. This operation preserves the adjacency information and guarantees unique mapping via VF-2 excluding the hydrogen-symmetries. Furthermore, the compression accelerates the matching process since the graph size is approximately halved.

After the identification of chemically correct mappings, our implementation generates an annotated reaction SMILES. The returned SMILES contains a corresponding numbering of mapped atoms in the educts and products. The following annotated reaction SMILES is the mapping result of the reaction R5 above. In addition, we insert ITS participated atoms (displayed in gray) into the reaction SMILES output. The ITS encoding exhibits bond changes that occurred along the reaction[1] represented by (+) for bond formation, (-) for bond breakage and (=) for non-changeable bonds. Note that ITS hydrogen atoms are now shown in the annotated reaction SMILES, since they are mapped as part of the ITS.

[1]The ITS encoding inserted into the mapped reaction is following the used ITS string encoding.

```
[C:8][C:7][O:6][C:4]([C:3])=[O:5].[H:10][O:1].[H:9][Cl:2]
>[O:1]+[C:7]-[O:6]+[H:9]-[Cl:2]+[H:10]->
[C:3][C:4]([O:1])=[O:5].[H:10][Cl:2].[H:9][O:6][C:7][C:8]
```

Figure 4.6 visualises this atom mapping result as molecule graphs together with the ITS subgraph between educts and products molecule graph. The figure also shows hydrogen atoms participated in the ITS.

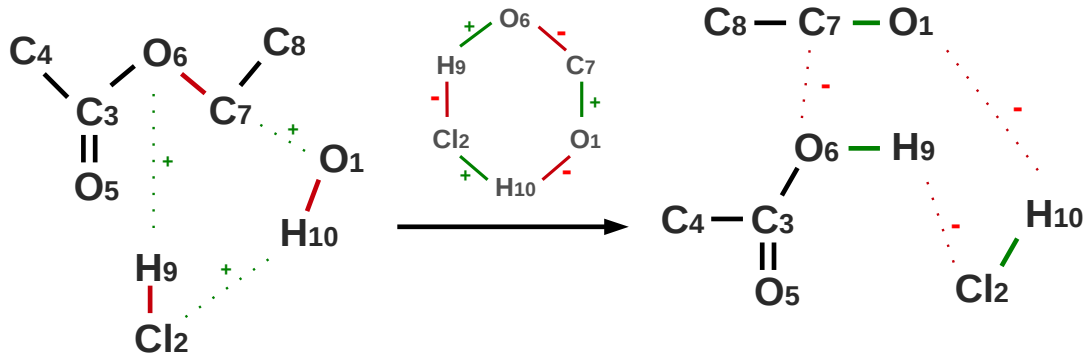

FIGURE 4.6: Mapping result of the reaction R5. Bonds which broken are in red, newly formed bonds are in green.

The workflow in figure 4.7 sketches the implementation details of the generic atom mapping framework.

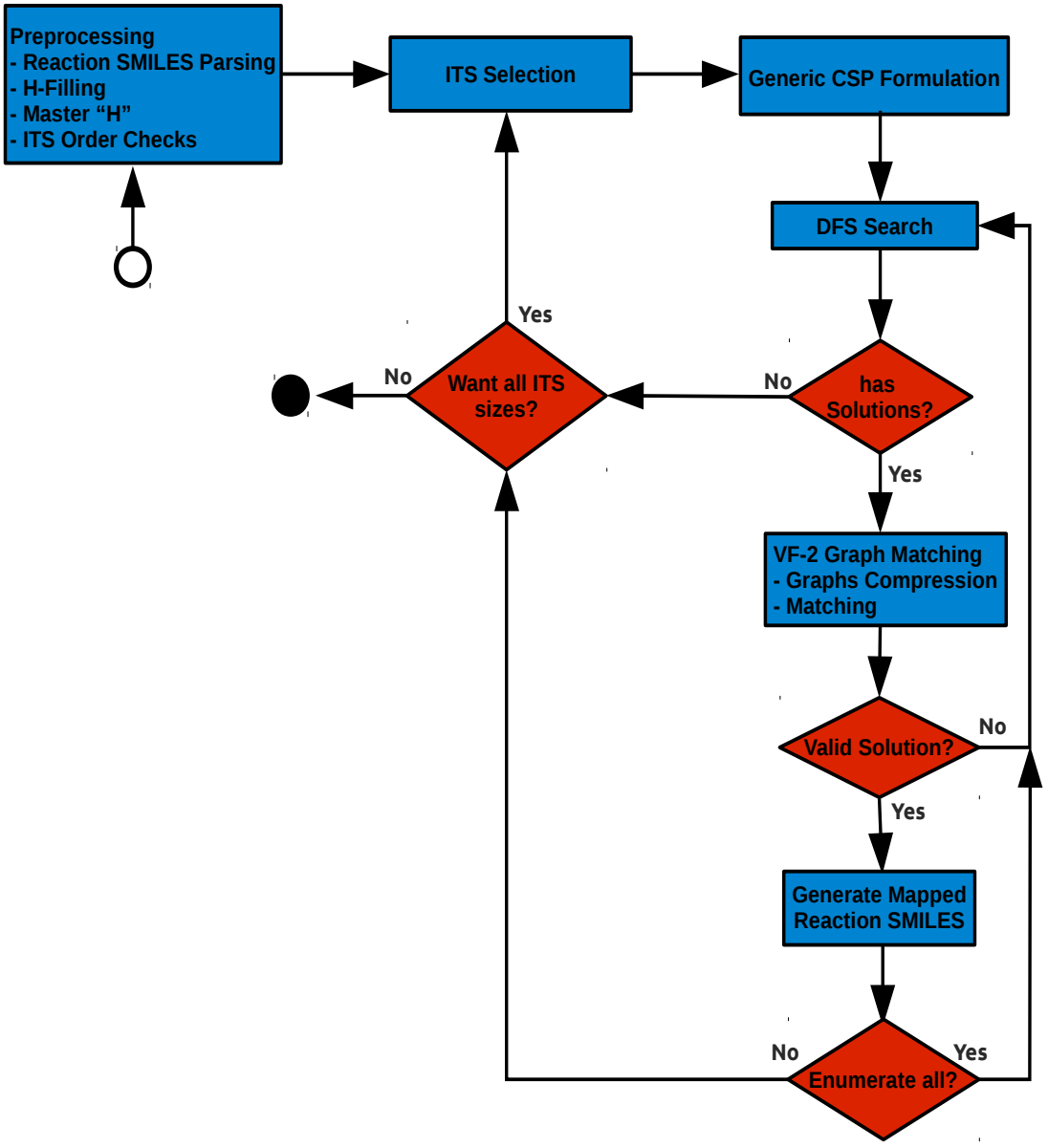

FIGURE 4.7: Workflow of the generic atom mapping framework. The open circle represents program's begin while the filled circle indicates the end.

Chapter 5

Tests and Evaluation

This chapter presents an evaluation of the previously mentioned CSPs and the generic atom mapping framework. We demonstrate here experimental results acquired by testing a number of chemical reactions.

5.1 Elementary Homovalent Reactions

In order to evaluate the performance of the constraint-based atom mapping, we selected several elementary homovalent and ambivalent reactions. The homovalent reactions R1 to R6 were taken from `http://www.imada.sdu.dk/~daniel/DM832-2012/assignment2/assign2-2012.html`. The rest of homovalent reactions is a collection from the KEGG LIGAND database [7]. Test reactions are listed in the table 5.1.

Each reaction was tested for increasing ITS ring size k using four CSP formulations: basic, extended, and full with bond valence conservation. We provide both the number of overall CSP solutions and correct mappings. The column "Overall" in the table 5.2 displays all chemically correct and incorrect ITS mappings and may contain symmetries, whereas column "Valid" contains overall atom mappings that are matched by the VF-2 algorithm. This column may also include equivalent mappings, since the symmetry exclusion procedure of the generic atom mapping framework is not applied here. The corresponding timings are given in seconds. For the extended CSP version, the precomputed ITS atoms are also shown. Table 5.2 reports test results.

ID	Educts	Products	Atoms
R1	2× C=C	C1CCC1	8
R2	2× C#N	N=CC#N	6
R3	C1C(O)CC(O)C(O)C1	3× C=CO	21
R4	CC, OC1C=CC=CC=1	C=C, OC(=C)C=CC=C	21
R5	O, Cl, CC(=O)OCC	Cl, OCC, CC(=O)O	19
R6	OP(=O)(O)OP(=O)(O)O, O	2× O=P(O)(O)O	16
R00009	2× OO	O=O, 2× O	8
R00013	2× C(=O)(C=O)O	C(=O)=O, C(C(=O)O)(C=O)O	14
R00018	2× C(CCN)CN	N, N(CCCCN)CCCCN	36
R00048	[CH](OC(=O)C[CH](C)O)(CC(=O)O)C, O	2× C[CH](CC(=O)O)O	30
R00059	N(C(=O)CCCCCN)CCCCCC(=O)O, O	2× C(CC(=O)O)CCCN	44
R00207	O=O, P(=O)(O)(O)O, CC(=O)C(=O)O	P(=O)(OC(=O)C)(O)O, C(=O)=O, OO	20

TABLE 5.1: Elementary homovalent reactions used for the evaluation of the approach. The educt and product molecules are given in SMILES notation [6]. The number of atoms in a reaction refers to the atom number after hydrogen filling.

			Solutions		Time in sec			
ID	k	CSP	Overall	Valid	1st Sol.	CSP	VF-2	Total
R1	4	Basic	1.424	2	0	0.02	0.17	**0.19**
		Ext.{4C}	16		0	0	0.01	**0.01**
		Full	16		0	0.01	0.01	**0.02**
		Val. Conserv.	16		0	0.01	0	**0.01**
R2	4	Basic	2	2	0	0	0	**0**
		Ext.{2C, N, H}	2		0	0	0	**0**
		Full	2		0	0	0	**0**
		Val. Conserv.	2		0	0	0	**0**
R3	6	Basic	220.776	2	0.03	7.11	49.63	**56.74**
		Ext.{6C}	96		0.01	1.62	0.05	**1.67**
		Full	96		0	2.07	0.04	**2.11**
		Val. Conserv.	96		0	5.03	0.08	**5.11**
R4	6	Basic	385.960	2	19.23	10.71	89.08	**99.79**

		Ext.{4C}	32		0.62	1.04	0.01	**1.05**
		Full	140.490		0	8.76	32.05	**40.81**
		Val. Conserv.	8.160		0	13.71	2.01	**15.72**
R5	4	Basic	1.804	4	0	0.03	0.33	**0.36**
		Ext.{2O}	8		0	0.04	0	**0.04**
		Full	1.512		0.01	0.04	0.31	**0.35**
		Val. Conserv.	232		0.01	0.16	0.04	**0.2**
	6	Basic	26.414	46	0.08	0.35	5	**5.35**
		Ext.{2O}	1.594		0.04	0.33	0.32	**0.65**
		Full	20.252		0.06	0.77	3.81	**4.58**
		Val. Conserv.	5.808		0.07	1.76	1.13	**2.89**
	8	Basic	206.962	298	0.76	3.77	38.3	**42.07**
		Ext.{2O}	58.294		0.66	3.15	12.03	**15.18**
		Full	155.562		0.42	6.46	6.46	**12.92**
		Val. Conserv.	89.702		0.58	14.85	18.37	**33.22**
R6	4	Basic	1.116	2	0.08	0.01	0.22	**0.23**
		Extended {2O}	36		0.01	0.02	0.01	**0.03**
		Full	1.044		0.1	0.06	0.2	**0.26**
		Val. Conserv.	972		0.13	0.05	0.26	**0.31**
	6	Basic	16.512	14	0	0.21	3.2	**3.41**
		Ext.{2O}	1.212		0.01	0.14	0.31	**0.45**
		Full	15.000		0	0.56	2.88	**3.44**
		Val. Conserv.	10.790		0	1.03	2.26	**3.29**
	8	Basic	151.032	48	0.01	2.41	27.56	**29.97**
		Ext.{2O}	19.464		0.01	0.67	4.71	**5.38**
		Full	129.096		0.01	4.49	23.54	**28.03**
		Edge Valence	84.094		0.01	6.48	17.07	**23.55**
R00009	6	Basic	64	4	0	0.01	0.01	**0.02**
		Ext.{4O}	16		0	0	0.01	**0.01**
		Full	64		0	0.01	0.02	**0.03**
		Val. Conserv.	64		0	0	0.02	**0.02**
R00013	6	Basic	346	2	0.03	0.08	0.03	**0.11**
		Ext.{2C}	76		0.02	0.05	0.02	**0.07**
		Full	84		0.03	0.09	0	**0.09**
		Val. Conserv.	40		0.03	0.07	0	**0.07**
	8	Basic	408	2	0.07	0.15	0.07	**0.22**
		Extended {2C}	164		0.06	0.17	0.02	**0.19**
		Full	164		0.07	0.18	0.01	**0.19**

		Val. Conserv.	154		0.07	0.17	0.02	**0.19**
R00018	4	Basic	73.924	8	10.42	2.62	19.9	**22.52**
		Ext.{2N}	36		0.28	0.44	0.01	**0.45**
		Full	73.924		12.3	5.05	19.62	**24.67**
		Val. Conserv.	63.988		77.02	100.49	17.21	**117.7**
	6	Basic	14.209.240	104	6.18	338.2	4103.83	**4442**
		Ext.{2N}	13.584		0.18	22.61	4.69	**27.3**
R00048	4	Basic	26.178	2	0.1	1.44	6.05	**7.49**
		Ext.{2O}	24		0.02	0.42	0.03	**0.45**
		Full	21.758		0.06	2.12	5.16	**7.28**
		Val. Conserv.	3.960		0.16	10.44	1	**11.44**
	6	Basic	2.685.708	20	0.1	88.76	666.57	**755.33**
		Ext.{2O}	6.946		0.05	16.01	1.95	**17.96**
		Full	2.065.636		0.11	151.65	505.18	**656.83**
		Val. Conserv.	422.637		0.15	835.21	118.57	**953.78**
R00059	4	Basic	194.210	1	0.34	9.45	63.15	**72.6**
		Ext.{H, C, N, O}	4		0.03	2.08	0.01	**2.09**
		Full	171.082		0.4	15.74	69.5	**85.24**
		Val. Conserv.	4.925		3.71	124.4	1.91	**126.31**
R00207	8	Basic	20.640	6	0.02	1.11	4.05	**5.16**
		Ext.{C, 4O}	24		0.01	0.56	0.02	**0.58**
		Full	24		0.02	0.1	0	**0.1**
		Val. Conserv.	24		0.02	0.13	0.01	**0.14**

TABLE 5.2: Evaluation of the reactions from table 5.1 using different CSPs.

The atom mapping approach finds the first mapping for most homovalent elementary reactions within milliseconds. Additional constraints within the extended CSP formulation significantly increase the performance of the approach. This becomes clear when considering the timings for overall solution enumeration, providing that the extended CSP produces much less ITS candidates (column "Overall"). Since the consumption time of the VF-2 algorithm is about linear in the number of ITS candidates to test, we gain a speed up of the overall approach. Testing the reaction R00018 for $k = 6$ using basic CSP reveals a huge ITS candidate number 14.209.240 with a respective overall time of 4.442 sec $\approx$ 74 min. However, the application of the extended CSP reduces this

greatly to 13.584 candidates and locates all mappings within 27.3 sec. Note, it is not true that the shortest ITS cycle size is always chemically correct. Other (larger) ITS layouts are often possible and have to be considered. However, for R00009 the only appropriate ring size is $k = 6$ and for reaction R00207 is $k = 8$.

When evaluating the full and the full edge valence conservation CSPs, we notice the increase in CSP size and accordingly in the propagation and search effort. This is because full CSP and edge valence conservation CSP are constraining all atoms in the reaction to preserve atom label, node degree, and bond valence information. The efficiency of the VF-2 graph matching approach does not compensate the large CSP size in this case. Only for the reaction R00013 with ring sizes ($k = 6$, $k = 8$), edge valence conservation CSP was comparably fast to the extended CSP. However it does not have a considerable impact on the performance.

The strength of the extended CSP comes from the precomputed list of local neighbourhoods to be part of the ITS candidate, which sometimes covers the whole ITS as in the reaction R00059. On average, this list comprises about the half of the ITS resulting in an enormous reduction in the number of ITS candidates and an impressive positive impact on the performance. Still it is possible to reduce the number of invalid ITS candidates through additional symmetry breaking provided by the generic atom mapping framework. Due to the inefficiency of the constraints used in full and valence conservation formulation, we do not integrate them in the generic atom mapping approach. The incorporation the precomputed local neighbourhood list from the extended CSP into the generic framework is sufficient to enumerate atom mappings efficiently.

5.2 Generic Framework vs. Extended CSP

So far we have evaluated test reactions using different CSP formulations. The following table 5.3 reports mapping and timing results when employing the generic atom mapping framework. We present here a comparison between the generic formulation and the extended CSP formulation in terms of the number of solution candidates, correct mappings, and the overall time. We chose the extended formulation to compare with, since it the most efficient CSP variant among the different CSP extensions. Only the

results of testing KEGG LIGAND reactions are reported in the following table. Column "Time" refers to the overall time of the CSP and VF-2 matching for the generic framework and for the extended CSP respectively.

		Generic			Extended		
ID	k	Overall Sol.	Valid Sol.	Time	Overall Sol.	Valid Sol.	Time
R00009	6	1	1	**0**	16	4	**0.01**
R00013	6	19	1	**0.13**	76	2	**0.07**
	8	41	1	**0.6**	164	2	**0.19**
R00018	4	2	1	**0.15**	36	8	**0.45**
	6	1438	13	**4.5**	13.584	104	**27.3**
R00048	4	8	2	**0.24**	24	2	**0.45**
	6	1792	20	**5.93**	6.946	20	**17.96**
R00059	4	1	1	**1.05**	4	1	**2.09**
R00207	8	1	1	**1.33**	24	6	**0.58**

TABLE 5.3: Evaluation of the KEGG LIGAND reactions from table 5.1 using generic atom mapping framework compared to the extended CSP. Timings are given in seconds.

Comparing with the extended CSP, the generic framework produces much less ITS candidates to be checked by the VF-2 procedure, which is reflected in the column "Generic Overall Sol.". The symmetry elimination and connected components coverage used in the generic framework contribute to this reduction, so that only non-symmetric ITS mappings in which all molecules are covered, are determined. As already expected based on the results from other approaches [23], most of the reactions shows a single mechanism (column "Generic Valid Sol.") for the smallest valid cycle size. The timings reveal a comparable performance (see columns "Generic Time" and "Extended Time").

5.3 Elementary Ambivalent Reactions

We evaluate here the generic atom mapping framework for elementary ambivalent reactions presented in the table 5.4.

Table 5.5 shows mapping and timing results for the ambivalent reactions from the table above. For each reaction, the corresponding ITS string encoding is given.

ID	Educts	Products	Atoms
AR1	`Cl[C--]Cl.C=C`	`ClC1(Cl)CC1`	9
AR2	`O=[S--]=O.C=CC=C`	`O=S1(=O)CC=CC1`	13
AR3	`[Cl][Si](C)(C)C.[O-][S+](C)C`	`C[S+](C)O[Si](C)(C)C.[Cl-]`	24
AR4	`[O-][NH2+]CC=C`	`NOCC=C`	12

TABLE 5.4: Elementary ambivalent reactions used to evaluate the approach. The number of atoms in a reaction refers to the atom number after hydrogen filling.

ID	k	ITS-Encoding	Solutions		Total Time
			Overall	Valid	
AR1	3	`[+2]+[0]-[0]+`	1	1	**0**
AR2	5	`[+2]+[0]-[0]+[0]-[0]+`	3	1	**0**
AR3	3	`[+1]+[0]-[-1]=`	1	1	**0**
AR4	5	`[+1]+[0]-[0]+[0]-[-1]=`	7	1	**0**

TABLE 5.5: Evaluation of the ambivalent reactions from table 5.4 using generic atom mapping framework. Timings are given in seconds.

In case of odd rings, the generic CSP locates all mappings within fractions of seconds for all different ITS layouts shown in the table 5.5. Note that, the symmetric assignments of the ITS are countered here by the ITS-specific order constraint mentioned in 4.4.2. However, the ambivalent layouts supported by the generic framework (sections 3.4.2.1, 3.4.2.2 and 4.1) show no symmetric matches in itself such that actually no order constraint is needed in this case. The strength of the generic CSP here comes from the propagation of the oxidation state change (charge change) for the atoms that get charged. This poses a very strong constraint for the ambivalent ITS identification resulting in few ITS candidates and consequently in a good performance. The approach selects the suitable ITS layout based on the provided reaction input.

Chapter 6

Conclusions

6.1 Conclusions

We have implemented here the first constraint programming approach presented in [2] to identify atom mappings for elementary homovalent reactions. We extended this approach to cover elementary ambivalent reactions which are more frequent in chemistry. To ensure the chemical feasibility of the mapping, the approach depends solely on the determination of the cyclic ITS structure within the mapping procedure. Chemical feasibility of the mapping is not guaranteed by standard approaches that attempt to solve e.g. Maximum Common Edge Subgraph Problems [19].

After formulating and evaluating different CSP models, we came up with the generic atom mapping CSP, which provides a universal encoding able to describe almost all possible elementary ITS layouts. This avoids the formulation of several separate CSPs for each corresponding ITS layout and is easy extendible to incorporate new layouts. The formulation of the CSP using only the atoms involved in the ITS results in a very small CSP that can be solved efficiently. Thus, it filters the ITS candidates for the subsequent, computationally more expensive graph matching approaches. The ITS-centered approach is particularly appealing when additional information on the ITS can be derived from the input. The lower bound precomputation of the ITS-involved atoms through the local neighbourhood constraint is used for this purpose and shows an impressive effect on the performance. Additionally we apply advanced symmetry

breaking strategies and thus can enumerate distinct mechanisms of a reaction for a given ITS cycle size.

The generic atom mapping framework enables through the predefined set of ITS layouts to draw conclusions, which reaction can be described though which mechanisms? In this sense, statistics can be made regarding those mechanisms that seem to be most often, etc. Furthermore, the atom mapping framework allows to figure out the reasons for those reactions that could not be mapped, say some layouts are missing in the framework or reactions with combined ITS layouts.

Constraint programming has proven to be a suitable and an efficient approach for solving chemoinformatics problems such as the atom mapping problem. It offers the expressiveness and the flexibility in formulating and solving chemical-driven tasks.

The results from this thesis were partially reported in [5]: "Atom Mapping with Constraint Programming" *in Proc. of the 19th International Conference on Principles and Practice of Constraint Programming (CP 2013)*, 2013.

6.2 Future Work

As future work we would like to use the generic atom mapping framework both as stand alone tool as well as via a web front end including a visual depiction of the atom mappings. The atom mapping framework will allow to discover unknown reaction mechanisms. The available layout set will grow gradually to involve newly discovered ITS arrangements. Additionally this work provides the core platform to determine atom mappings for complex reaction mechanisms. The determination of elementary ITS could be extended to non-elementary transition states that are based on two or more elementary ITSs.

The analysis of the metabolic networks is based on reaction mappings [11], since atom maps are used to perform consistency checks on pathway data. In other words, the detection of correct routes within the atom flow network implies the chemical validity of the atom maps. This can be ensured with the given approach. Furthermore, the atom mapping approach could be used to generate chemical graph grammar rules that will be used in the GGL framework [36]. This would allow to expand the chemical space and

according reactions network where molecular graph rewrite directly provides the atom flow information within the network.

Bibliography

[1] Daylight. Chemical information systems, inc. `http://www.daylight.com`. Accessed: February, 2013.

[2] Martin Mann, Heinz Ekker, Peter F. Stadler, and Christoph Flamm. Atom mapping with constraint programming. In *Proceedings of the Workshop on Constraint Based Methods for Bioinformatics (WCB 2012)*, page 7, 2012.

[3] Jakob Meisenheimer. Über eine eigenartige Umlagerung des Methyl-allyl-anilin-N-oxyds. *Chemische Berichte*, 52:1667–1677, 1919.

[4] J. B. Hendrickson. Comprehensive system for classification and nomenclature of organic reactions. *J Chem Inf Comput Sci*, 37:852–860, 1997.

[5] Martin Mann, Feras Nahar, Heinz Ekker, Rolf Backofen, Peter F. Stadler, and Christoph Flamm. Atom mapping with constraint programming. In *Proceedings of the 19th International Conference on Principles and Practice of Constraint Programming (CP 2013)*, LNCS, page 16. Springer-Verlag, 2013. Accepted for publication.

[6] D. Weininger. SMILES, a chemical language and information system. 1. Introduction to methodology and encoding rules. *J. Chem. Inf. Comp. Sci.*, 28(1):31–36, 1988. doi: 10.1021/ci00057a005.

[7] M. Kanehisa, S. Goto, Y. Sato, M. Furumichi, and M. Tanabe. KEGG for integration and interpretation of large-scale molecular data sets. *Nuc. Acids Res.*, 40 (Database issue):D109–14, 2012. doi: 10.1093/nar/gkr988.

[8] W. Wiechert. ^{13}C metabolic flux analysis. *Metabolic Engineering*, 3:195–206, 2001.

[9] L. Chen and J. Gasteiger. Knowledge discovery in reaction databases: Landscaping organic reactions by a self-organizing neural network. *J Am Chem Soc*, 119:4033–4042, 1997.

[10] M. Arita. The metabolic world of Escherichia coli is not small. *Proc. Natl. Acad. Sci. USA*, 106:1543–1547, 2004.

[11] T. Blum and O. Kohlbacher. Using atom mapping rules for an improved detection of relevant routes in weighted metabolic networks. *Journal of Computational Biology*, 15:565–576, 2008.

[12] A. P. Heath, G. N. Bennett, and Lydia E. Kavraki. An algorithm for efficient identification of branched metabolic pathways. *Journal of Computational Biology*, 18(11):1575–1597, November 2011.

[13] I. Ugi, J. Bauer, Kl. Bley, A. Dengler, A. Dietz, E. Fontain, B. Gruber, R. Herges, M. Knauer, K. Reitsam, and N. Stein. Computer-assisted solution of chemical problems-the historical development and the present state of the art of a new discipline of chemistry. *Angew. Chem. Int. Ed. Engl.*, 32:201–2267, 1993.

[14] C. Jochum, J. Gasteiger, and I. Ugi. The principle of minimum chemical distance (PMCD). *Angew. Chem. Int. Ed.*, 19:495–505, 1980.

[15] Ivan Gutman, Dus;ica Vidovic, and Ljiljana Popovic. Graph representation of organic molecules Cayley's plerograms vs. his kenograms. *J. Chem. Soc., Faraday Trans.*, 94:857–860, 1998.

[16] J. W. Raymond and P. Willett. Maximum common subgraph isomorphism algorithms for the matching of chemical structures. *J. Computer-Aided Mol. Design*, 16:521–33, 2002.

[17] M. Heinonen, S. Lappalainen, T. Mielikäinen, and J. Rousu. Computing atom mappings for biochemical reactions without subgraph isomorphism. *J. Comp. Biol.*, 18:43–58, 2011.

[18] H.-C. Ehrlich and M. Rarey. Maximum common subgraph isomorphism algorithms and their applications in molecular science: a review. *WIREs Comput Mol Sci*, 2011. doi:10.1002/wcms.5.

[19] T. Akutsu. Efficient extraction of mapping rules of atoms from enzymatic reaction data. *J. Comp. Biol.*, 11:449–62, 2004.

[20] R. Körner and J. Apostolakis. Automatic determination of reaction mappings and reaction center information. 1. The imaginary transition state energy approach. *J. Chem. Inf. Mod.*, 48:1181–1189, 2008.

[21] J. Apostolakis, O. Sacher, R. Körner, and J. Gasteiger. Automatic determination of reaction mappings and reaction center information. 2. Validation on a biochemical reaction database. *J. Chem. Inf. Mod.*, 48:1190–1198, 2008.

[22] L. Felix and G. Valiente. Efficient validation of metabolic pathway databases. In *In Proc. 6th Int. Symp. Computational Biology and Genome Informatics*, 2005.

[23] E.L. First, C.E. Gounaris, and C.A. Floudas. Stereochemically consistent reaction mapping and identification of multiple reaction mechanisms through integer linear optimization. *J. Chem. Inf. Model.*, 52(1):84–92, 2012. doi: 10.1021/ci200351b.

[24] S Fujita. Description of organic reactions based on imaginary transition structures. 1. Introduction of new concepts. *J. Chem. Inf. Comput. Sci.*, 26:205–212, 1986.

[25] Rainer Herges. Organizing principle of complex reactions and theory of coarctate transition states. *Angewante Chemie Int Ed*, 33:255–276, 1994.

[26] James Dugundji and Ivar Ugi. An algebraic model of constitutional chemistry as a basis for chemical computer programs. *Topics Cur. Chem.*, 39:19–64, 1973.

[27] Shinsaku Fujita. Description of organic reactions based on imaginary transition structures. 3. Classification of one-string reactions having an odd-membered cyclic reaction graph. *Journal of Chemical Information and Computer Sciences*, 26(4): 224–230, 1986.

[28] Roman Bartak. Online guide to constraint programming. `http://ktiml.mff.cuni.cz/~bartak/constraints/constrsat.html`. Accessed: February, 2013.

[29] Gecode. Generic constraint development environment. `http://www.gecode.org`. Accessed: May, 2013.

[30] Shinsaku Fujita. Description of organic reactions based on imaginary transition structures. 2. Classification of one-string reactions having an even-membered cyclic reaction graph. *Journal of Chemical Information and Computer Sciences*, 26(4): 212–223, 1986.

[31] L.P. Cordella, P. Foggia, C. Sansone, and M. Vento. A (sub)graph isomorphism algorithm for matching large graphs. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 26(10):1367–72, 2004.

[32] L.P. Cordella, P. Foggia, C. Sansone, and M. Vento. Performance evaluation of the VF graph matching algorithm. In *Proceedings of the 10th International Conference on Image Analysis and Processing*, ICIAP '99, page 1172. IEEE Computer Society, 1999.

[33] Guido Tack. *Constraint Propagation - Models, Techniques, Implementation.* PhD thesis, Saarland University, Germany, 2009. URL `http://www.gecode.org/paper.html?id=Tack:PhD:2009`.

[34] Heinz Ekker. Automatic extraction of graph rewrite rules from biochemical reactions. Diploma thesis, Institute for Theoretical Chemistry, University of Vienna, 2010.

[35] Martin Mann. *Computational Methods for Lattice Protein Models.* PhD thesis, Albert-Ludwigs-University Freiburg, June 2011.

[36] M. Mann, H. Ekker, and C. Flamm. The graph grammar library - a generic framework for chemical graph rewrite systems. In Keith Duddy and Gerti Kappel, editors, *Theory and Practice of Model Transformations, Proc. of ICMT 2013*, volume 7909 of *LNCS*, pages 52–53. Springer, 2013. ISBN 978-3-642-38882-8. doi: 10.1007/978-3-642-38883-5_5. Extended abstract at ICMT, long version at arXiv `http://arxiv.org/abs/1304.1356`.

Selbstständigkeitserklärung

Hiermit erkläre ich, dass ich die vorliegende Arbeit selbstständig und nur unter Verwendung der angegebenen Quellen und Hilfsmittel angefertigt habe.

Ort, Datum:

Unterschrift:
