Reasoning about Chemical Reactions using the Situation Calculus

Arman Masoumi
Ryerson University
Department of Computer Science
Toronto, Ontario, Canada
{arman.masoumi@ryerson.ca}

Mikhail Soutchanski
Ryerson University
Department of Computer Science
Toronto, Ontario, Canada
{mes@cs.ryerson.ca}

Abstract

We explore applicability of the *situation calculus*, the well-known logical framework developed in Artificial Intelligence for representation of dynamic systems, to the task of representing knowledge about processes, actions and events in the natural sciences. In this paper, we concentrate on a case study in the area of organic chemistry. More specifically, we adapt the situation calculus to the task of automating organic synthesis planning on a qualitative level, where the objective is to identify a chain of chemical reactions transforming the given initial molecules into the desired goal molecule. We present two approaches for reasoning about reactions in organic chemistry: a "micro" approach and a "macro" approach. The "micro" approach is a low level approach that explicitly represents the most elementary interactions between molecules during a single chemical reaction, namely the splitting and forming of bonds between atoms. In contrast, the "macro" approach is a higher level approach that treats each chemical reaction (a set of splits and formation of bonds) as an elementary action. Both approaches are implemented in PROLOG. Declarative heuristics are defined to reduce the search space and help the program to find the correct synthesis routes more quickly. We hope that the lessons learned from our successful case study can have discovery potential in other bio-medical sciences. We discuss briefly how the proposed approaches can contribute to solving other research problems and to communicating pathways.

1 Introduction

In Artificial Intelligence (AI), the *situation calculus* (SC) is one of the most popular and well-known logical formalisms for representation of and reasoning about the effects of actions and events. The situation calculus was first introduced by John McCarthy in the 1960s. The version of SC that we use here follows (Reiter 2001). A recent new fragment (Yehia and Soutchanski 2012) of SC enables seamless integration of traditional ontologies with reasoning about actions. More specifically, any OWL2 ontology can be used to annotate logical axioms specifying effects of actions. The general SC has been already applied in several areas, but, surprisingly, to the best of our knowledge, not for the purposes of automating discovery in natural sciences. We hypothesize that, as a well-explored logical framework with known reasoning mechanisms, SC offers significant potential for representation and reasoning about dynamics in the sciences (including biological pathways and physiological processes). In our paper, we concentrate on a case study related to the computer-assisted organic synthesis (CAOS) problem. This problem is important in drug discovery and in the field of organic chemistry in general. We show that this problem can be conveniently formulated as a planning problem in SC. We consider two different approaches to axiomatizing chemical reactions in SC: a "*micro*" approach and a "*macro*" approach. We argue that the planning problem can be made computationally feasible by using declarative heuristics that reduce the search space significantly. We have successfully implemented both approaches in PROLOG. Our program can compute a sequence of chemical reactions capable of producing a desired target molecule from initially available molecules. Using our PROLOG program we can successfully solve a variety of instances of the CAOS problem. We report run-time data collected when the planning problem is solved with or without declarative heuristics for several target molecules. Our data show that the heuristics help the program to find the correct synthesis route more quickly. Our research can be useful not only for solving the organic synthesis planning problem, but also for solving other practical computational problems. Also, since our research can be applied in other areas beyond organic chemistry, we hope that the lessons learned from our successful case study can have the potential of facilitating discovery in the biomedical sciences. In Section 6, we argue briefly why the proposed approaches can facilitate accomplishments in the biomedical sciences.

2 Background

To make our paper readily comprehensible we review both the situation calculus and the basics of organic chemistry.

The situation calculus (SC) is a predicate logic language for axiomatizing dynamic worlds. In recent years, it has been considerably extended beyond the original language to include stochastic actions, concurrency, continuous time, etc, but in all cases, its basic ingredients consist of *actions* a, *situations* s and *fluents* (Reiter 2001). *Actions*, $A(\vec{x})$, where $\vec{x}$ is a tuple of distinct *object* variables, are first order logic (FOL) terms consisting of an action function symbol A and its arguments $\vec{x}$. A *situation* is a first-order term denoting a sequence of actions. Such sequences are represented using a binary function symbol do: $do(a, s)$ denotes the sequence resulting from adding an action term a to the sequence s. The special constant S_0 denotes the *initial situation*, namely the empty action sequence. Every situation refers uniquely to a sequence of actions. SC includes the predicate $Poss(a, s)$ to characterize actions a that are possible to execute in s, and the predicate $s_1 \preceq s_2$ to specify precedence between situations s_1 and s_2. As usual, we say that a situation calculus formula $\psi(s)$ is *uniform* in s, if s is the only situation term mentioned in $\psi(s)$, the formula ψ has no occurrences of the predicates $Poss, \prec$, and has no quantifiers over situations. Fluents $F(\vec{x},s)$ are predicates with the last argument s being a situation. Their truth values may depend on actions in s.

A *basic action theory* (BAT) $\mathcal{D}$ is a set of axioms for a domain theory written in SC with the following five classes of axioms to model actions and their effects (Reiter 2001). (We give examples of axioms in the subsequent sections.) In axioms, all free variables (typically starting with lower case letters) are implicitly universally ($\forall$) quantified at front. Read the symbol $\wedge$ as 'and', $\vee$ as 'or', $\exists$ as exists, $\neg$ as 'not'.

Action precondition axioms (PA) $\mathcal{D}_{ap}$**:** There is one axiom for each action term $A(\vec{x})$, with syntax $Poss(A(\vec{x}), s) \leftrightarrow \Pi_A(\vec{x}, s)$. Here, $\Pi_A(\vec{x}, s)$ is an uniform formula composed from fluents with free variables among $\vec{x}, s$. These are the preconditions of action A: A is possible if and only if (use the bi-conditional $\leftrightarrow$ for iff) the condition $\Pi_A(\vec{x}, s)$ holds.

Successor state axioms (SSA) $\mathcal{D}_{ss}$**:** There is one axiom for each fluent $F(\vec{x}, s)$, with syntactic form $F(\vec{x}, do(a, s)) \leftrightarrow \Phi_F(\vec{x}, a, s)$, where $\Phi_F(\vec{x}, a, s)$ is a formula uniform in s with free variables among $a, s, \vec{x}$. Each SSA is as follows:

$$F(\vec{x}, do(a, s)) \leftrightarrow \bigvee_i a = \mathit{PosAction}_i(\vec{x}) \wedge \gamma_i^+(\vec{x}, s) \vee$$
$$F(\vec{x}, s) \wedge \neg\big(\bigvee_j a = \mathit{NegAction}_j(\vec{x}) \wedge \gamma_j^-(\vec{x}, s) \big),$$

where $\mathit{PosAction}_i$ is an action that makes the fluent F true and $\gamma_i^+(\vec{x}, s)$ is the formula expressing a context in which this positive effect can occur; similarly, $\mathit{NegAction}_j$ is an action that can make the fluent F false if the formula $\gamma_j^-(\vec{x}, s)$ holds in s. If the executed action a is none of these, then the truth value of F remains unchanged (a has no effect). SSAs characterize the truth values of the fluent F in the next situation $do(a, s)$ in terms of fluents in the situation s and they represent non-effects of actions compactly.

Unique name axioms for actions and object constants $\mathcal{D}_{una}$**:** These state that the actions of the domain and the constants are pairwise unequal.

Initial theory$\mathcal{D}_{S_0}$**:** Specifies initial (incomplete) knowledge. This is a set of sentences whose only situation term is S_0.

Foundational axioms Σ**:** Sentences about the relation $s_1 \preceq s_2$ of precedence between situations s_1 and s_2 (Reiter 2001).

Below we summarize basic notions from aliphatic organic chemistry. (However, SC can be also used to reason about reactions of compounds with rings, such as benzene.) Molecules are formed by bonded atoms. Chemical valence is described as the number of bonds an atom can form. For example, a carbon atom can form four bonds, and it can also form at most two double bonds, e.g., in carbon dioxide CO_2, i.e., it has valence number of 4, while an oxygen atom has valence 2 and bonds with at most two atoms, e.g., in water H_2O; hydrogen has valence 1. Molecules are categorized into *chemical classes* based on their constitutive *functional groups*. Functional groups are specific groups of atoms within a molecule that are responsible for chemical characteristics of the molecule. The molecules that have the same functional groups behave similarly in chemical reactions. For example, *alkanes*, *alcohols* and *esters* are some of the well-known chemical classes that display unique behaviors in reactions. The main functional groups of these chemical classes are *alkyl*, *hydroxyl* and *ester*, respectively. Alkyls, usually denoted by R, are chemical compounds that consist solely of acyclic single bonded (univalent) carbon and hydrogen atoms such that all atoms have saturated bonds except for one carbon atom that can react (alkyls can exist as a straight chain of carbons or as branched chains of carbon atoms, known as isomers). For example, *methyl* CH_3- and *ethyl* $CH_3{-}CH_2$- are alkyls. A hydroxyl group OH- is an oxygen atom O that is bound with a hydrogen atom H. The ester functional group has the form COO$-$R, where R is an alkyl. Alkanes are compounds in which the alkyl functional group bonds with a hydrogen atom. For example, *methane* CH_4 is an alkane. An alcohol R$-$OH is a compound in which a hydroxyl functional group -OH is bound to a carbon atom in an alkyl R: for instance, *methanol* $CH_3{-}OH$ and *ethanol* $CH_3{-}CH_2{-}OH$ are alcohols. Esters are compounds of the form R$-$COO$-$R$'$, where both R and R$'$ are alkyls, and an oxygen atom has a double bond with carbon. Primary *alkyl halide* is R$-$X, where X is a halogen (e.g., F, Cl, Br, I).

A *moiety* (i.e., a portion) is generally used to signify part of a molecule that may include either a whole functional group or part of a functional group as substructures. For example, an ester R$-$COO$-$R$'$ has an ester functional group (COO$-$R$'$) and is composed of an *alkoxy* moiety (-OR$'$) and an *acyl* moiety (RCO-). Another example is alcohol R$-$OH which is composed of a hydroxyl moiety OH and an alkyl moiety R.

Computer-assisted organic synthesis aims to use computers to help chemists in the process of designing multistep synthesis of organic compounds. It has been an active area of research for a long time (Corey and Wipke 1969; Vléduts 1963). The 1990 Nobel Prize in Chemistry was awarded to E.J. Corey for the "development of the theory and methodology of organic synthesis". A variety of systems have been developed that help to find a sequence of chemical reactions synthesizing a target compound (Corey and Cheng 1995; Chen 2006; Todd 2005; Cook et al. 2012). It is interesting to realize that the technique of *retrosynthetic analysis* popular in CAOS literature resembles the technique of *regression* invented independently in AI (Waldinger 1975) and used extensively in SC for reasoning about the effects of actions (Reiter 2001). We hope that our paper will stimulate the discovery of other links and analogies with AI that will help to advance research in organic chemistry and related sciences.

3 Representing Chemical Reactions in SC

It is common to think of molecules as graphs, where the vertices of the graph represent the atoms forming the molecule and the edges of the graph represent the chemical bonds between the atoms. Fujita introduces Imaginary Transition Structure (ITS), a formalism to model chemical reactions (Fujita 2001; 1986). A somewhat related approach was developed by G.E. Vladutz, who defined "superimposed reaction graphs " and, by omitting atoms, which do not participate in bond alterations, a sub-graph called "superimposed reaction skeleton graph" (Vléduts and Geivandov 1974). In ITS, three kinds of bonds are defined depending on the role they play in a chemical reaction: *out*-bonds (bonds that exist only in the starting stage, but disappear after the reaction), *in*-bonds (bonds that exist only in the product stage, but not before the reaction), and *par*-bonds (bonds that exist in both starting and product compounds; these remain unchanged). This definition of bonds can be matched well in SC, if we introduce a fluent for bonds. When writing SSAs for bonds, we consider the bonds that will be formed (the fluent becomes true), the bonds that will be split (the fluent becomes false), and the bonds that do not change as a result of an action. Thus, we seek to write *general axioms* representing the effects of a class of chemical reactions by using the ITS definition for chemical reactions.

There are two approaches to writing axioms for a chemical synthesis design problem using SC: one using micro actions, the other using macro actions. The differences between the two approaches and their requirements will be discussed later. Here we explain what they have in common.

When solving an organic synthesis problem, there needs to be a set of chemical compounds to initiate the chemical reactions. This set of initial molecules is defined in the initial theory (D_{S_0}), which can be incomplete (if our initial knowledge is incomplete). The object constants (typically starting with a capital letter) in our axioms represent the physical atoms from the periodic table. Any number of atoms can be introduced using situation independent predicate symbols corresponding to the common names of the atoms. For example, $Hydrogen(H')$ means that the constant H' represents a hydrogen atom. Similarly, we can introduce predicates for a group of atoms. For example, $Halogen(x)$ is true if x belongs to the Halogen group in the periodic table (e.g., x might be Fluorine or Chlorine). Molecules are formed by bonds between the atoms; hence, a few fluents are introduced to determine whether an atom is bound to another in any situation. The fluent $Bond(x, y, s)$ is true if atom x has a single bond with atom y in situation s. Similarly, the fluent $DoubleBond(x, y, s)$ is true if there is a double bond between x and y in s. To represent bonds created by electrons shared within a ring we can introduce the fluent $ContinuousBond(x, y, s)$. The initial theory includes formulas about the bonds between the atoms corresponding to the starting stage molecules. For example, a water molecule H_2O in the initial situation S_0 can be represented as

$$\begin{aligned}&Hydrogen(H') \wedge Hydrogen(H'')\wedge\\ &\quad H' \neq H'' \wedge Oxygen(O)\wedge\\ &\qquad \forall x(Bond(x, O, S_0) \rightarrow (x = H' \vee x = H''))\wedge\\ &\qquad\quad \forall y(Bond(H'', y, S_0) \rightarrow y = O)\wedge\\ &\qquad\qquad \forall y(Bond(H', y, S_0) \rightarrow y = O).\end{aligned}$$

Additionally, in order to represent the reacting sub-graphs of each molecule, we need to characterize molecules as instances of *chemical classes* and identify their *functional groups*. To achieve this, several well-known chemical concepts are introduced as *abbreviations*. These concepts might include *alkyl*, *alcohol*, *ester*, *ether*, *water*, etc. We do not need SSAs for these abbreviations since they are definitions that can be expanded on demand. We allow only one atom as an object argument in each abbreviation. We call that atom the *key atom*. Having only one object argument in the abbreviations facilitates translating them to concepts in a *description logic* (Baader et al. 2003). The key atom of a molecule is nondeterministically picked out from a set of candidates that are generated based on identifying the moieties that compose the molecule. More specifically, the key atom is selected out of those atoms which belong to a moiety of the molecule and bond to another moiety, i.e., the key atom is one of the active end vertices bridging two moieties. For example, in alcohol R$-$OH, the carbon of the alkyl R and the oxygen of the hydroxyl group OH are two candidates for being the key atom. The abbreviation of alcohol with oxygen as the key atom would be this:

$$\begin{aligned}&Alcohol(o, s) \overset{def}{=} Oxygen(o) \wedge \exists^{=2}\, x(Bond(o, x, s))\wedge\\ &\quad \exists^{=1}\, h(Hydrogen(h) \wedge Bond(o, h, s))\wedge\\ &\qquad \exists^{=1}\, c(Alkyl(c, s) \wedge Bond(o, c, s)),\end{aligned}$$

where $\exists^{=1}$ ($\exists^{=2}$, respectively) is a counting quantifier saying that there exists exactly one (there are exactly two, respectively) entities for which quantified formula holds. Notice that in the abbreviation for alcohol we use another abbreviation, namely $Alkyl(c, s)$. The key atom of an alkyl is the carbon atom that is not saturated with hydrogen or other carbon atoms, but can form a bond. Similarly, by choosing the oxygen in a water molecule H_2O to be the key atom, the abbreviation for water would be:

$$\begin{aligned}&Water(o,s) \overset{def}{=} Oxygen(o) \wedge \exists^{=2}\, x(Bond(o, x, s))\wedge\\ &\quad \exists^{=2}\, h(Hydrogen(h) \wedge Bond(o, h, s)).\end{aligned}$$

Each abbreviation essentially describes all the atoms that are connected in the molecule and all the bonds among them.

Analogously, we introduce some abbreviations for formulas that characterize our desired target situation. The abbreviation $Goal(s)$ is used to characterize whether the sequence of actions represented by s is our goal. Goals depend on the planning instance we would like to solve. For example,

$$Goal(s) \overset{def}{=} (\exists o) Alcohol(o, s).$$

This formula states that our goal is to reach a situation s in which a key atom o belongs to an alcohol molecule.

The atoms, the initial theory, the abbreviations for chemical compounds and the goal states are defined in the same way in both micro and macro approaches. In addition to these, we need some PAs to specify when actions are possible. Also, SSAs are needed to determine the effect of the actions on bonds. However, each approach requires a different set of situation calculus actions. Consequently, we need a different set of PAs and a different set of SSAs for each approach. The next two subsections explain these approaches in detail.

3.1 Micro Approach

The micro approach considers the most elementary actions possible in the process of a chemical reaction, namely splitting and forming bonds between atoms. According to this approach, one chemical reaction translates into a sequence of actions. The micro approach needs only two actions, $splitBond$ and $formBond$, since any chemical reaction can be represented in terms of these elementary actions. For example, $splitBond(C_1, O_1, 1)$ means that a single bond between the atom C_1 and the atom O_1 splits, while $formBond(C_1, O_1, 1)$ means that a single bond forms between C_1 and O_1. These actions can also represent formation or splitting of multiple bonds. For example, $splitBond(C_1, O_1, 2)$ and $formBond(C_1, O_1, 2)$ represent splitting and formation of double bonds between C_1 and O_1.

Furthermore, in addition to the fluents representing bonds between atoms, we need to introduce some auxiliary fluents for keeping track of bonds between the key atoms of the intermediate compounds. The reason we need such fluents is that we are faced with intermediate molecules in this approach; that is, the imaginary sub-molecules that are created when a chemical reaction has started, but has not yet completed. It is easy to introduce these intermediate molecules if a reaction mechanism is known in advance. Otherwise, the axiomatizer may need to guess different alternative intermediates. We have to write SSAs for the effects of the actions on these fluents. An example of such SSAs would be:

$$\begin{aligned}&HydrogenSeparatedFromStrongAcid(h, do(a, s)) \leftrightarrow\\&\exists x(a = splitBond(h, x, 1) \lor a = splitBond(x, h, 1)) \land\\&\quad Strong\_acid(h, s) \land Hydrogen(h) \lor\\&HydrogenSeparatedFromStrongAcid(h, s) \land\\&\qquad \neg\exists x(\, a = formBond(h, x, 1) \lor\\&\qquad\qquad a = formBond(x, h, 1)\,).\end{aligned}$$

The fluent $HydrogenSeparatedFromStrongAcid(h, s)$ holds if h is a hydrogen ion (a proton) separated from a strong acid in the situation s. This fluent is true in $do(a, s)$ if and only if the most recent action a split the bond of the hydrogen ion from a strong acid, or if the hydrogen ion had been separated from a strong acid in previous situation s, and it did not form a new bond.

We have to axiomatize when it is possible to split or form bonds between atoms. There is one PA for each elementary action. Each PA covers all the possibilities for every pair of atoms at each intermediate stage of a reaction, e.g.,

$$Poss(splitBond(x,y,1),s) \leftrightarrow \Pi_{C\text{–}O} \lor \Pi_{H\text{–}O} \lor \cdots,$$

where $\Pi_{C\text{–}O}$, one of the disjunctive sub-cases, stands for:

$$\begin{aligned}&Carbon(x) \land Oxygen(y) \land Ester(x, s) \land Bond(x, y, s)\land\\&\quad \exists h, o'\ (Hydrogen(h) \land Oxygen(o')\land\\&\qquad\qquad Water(o', s) \land Strong\_acid(h, s)).\end{aligned}$$

This specifies a precondition for a carbon atom x to split its bond with its oxygen neighbor y. Specifically, it is possible to split the single bond between the key carbon atom and its single bonded oxygen neighbor in an ester molecule when both a molecule of water and a molecule of strong acid are present. To visualize this sub-case consider Figure 1. Here, we have an ethyl acetate, which is an ester, with a water molecule H_2O and a molecule of hydrochloric acid HCl, which is a strong acid. Therefore, all the conditions of this disjunctive sub-case $\Pi_{C\text{–}O}$ are satisfied. The $Ester(x, s)$ identifies the C atom and $Bond(x, y, s)$ identifies the neighbor O atom in the ester molecule in situation s (distinguished by the arrow pointing at them). There is also an oxygen atom o' that determines the presence of a water molecule, $Water(o', s)$, and a hydrogen atom h that determines a molecule of a strong acid, $Strong\_acid(h, s)$. Therefore, it is possible to split the bond between C and O.

The above example characterizes executability of an action initiating a chemical reaction, namely the ester and water reaction. But in the micro approach, we also need a precondition sub-case for every step of a chemical reaction. In other words, we need a sub-case for continuing an already started chemical reaction. These sub-cases take advantage of the auxiliary fluents that were introduced earlier to keep track of key atoms in intermediate compounds participating in each step. For example, in the situation resulting from executing the *splitBond*(*C*,*O*,1) action discussed above, there is a sub-molecule that was formerly part of the ester molecule. We identify it using the active carbon atom and represent it with the auxiliary $CarbonFromPartialEster(c,s)$ fluent. The following disjunctive sub-case $\Pi_{H\text{–}O}$ specifies when the second step of this reaction (splitting bond between H and O of a water molecule) is possible:

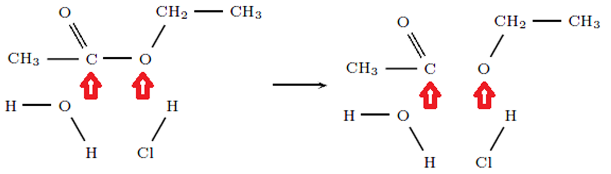


Figure 1: *splitBond* between carbon and oxygen atoms

$$\begin{aligned}&Hydrogen(x) \land Oxygen(y) \land Water(y, s)\land\\&\quad Bond(x, y, s) \land (\exists c)CarbonFromPartialEster(c, s).\end{aligned}$$

This sub-case states that it is possible to split the bond between a hydrogen atom and the oxygen atom of a water molecule if there is a carbon atom present that split from an ester. In Figure 2, you can see this second step of the chemical reaction. Here the carbon atom separated from an ester, $CarbonFromPartialEster(c, s)$, is identified with an arrow on the left hand side. Since there is such an atom and there is a molecule of water, it is possible to split the bond in the water molecule. The arrows on the right hand side of the figure point to the atoms in the water molecule that can split the bond.

The PA for the *formBond* action is similar. In general, in the micro approach, each of the two PAs for *splitBond* and *formBond* actions is a (potentially long) disjunction of conditions similar to those mentioned above.

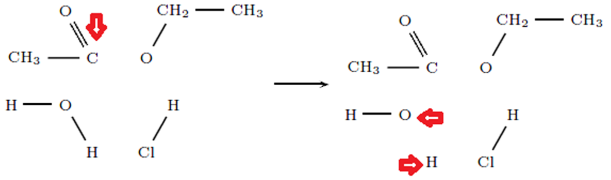


Figure 2: *splitBond* between hydrogen and oxygen of water

Finally, we need to write SSAs for the fluents that represent bonds between the atoms. These SSAs determine the direct effect of the actions on the $Bond(x, y, s)$ and $DoubleBond(x, y, s)$ fluents. Other bonds are similar.

The SSA for the $Bond$ fluent in the micro approach is quite simple, since there are only two actions. It is as follows:

$$\begin{aligned}&Bond(x, y, do(a, s)) \leftrightarrow (a = formBond(x, y, 1)\lor\\&a = formBond(y, x, 1)) \lor (Bond(x, y, s)\land\\&a \neq splitBond(x, y, 1) \land a \neq splitBond(y, x, 1)).\end{aligned}$$

This SSA states that two atoms have a single bond if and only if the last action added a single bond between them (the order of appearance of the atoms in the action is unimportant), or they already had a single bond and the last action did not split that bond.

To discuss the scalability of the axiomatization in the micro approach assume we have N reactions in our knowledge base (KB), where each reaction has at most M elementary steps (splitting and forming bonds). The PAs have syntax $Poss(act(x,y,z),s) \leftrightarrow \Pi_1 \vee \Pi_2 \vee \cdots$, where $act(x,y,z)$ is either $splitBond(x,y,z)$, or $formBond(x,y,z)$, and each Π_i (which represents a condition for the i-th pair of atoms) is a formula composed from fluent literals or abbreviations. Suppose each precondition sub-case in Π_i mentions at most K literals to specify when one elementary step between a pair of atoms can proceed in a reaction. Therefore, each PA will have on the order of $O(N \cdot M \cdot K)$ sub-cases. In practice, M and K are small constants, meaning there is an upper bound on the number of steps in reactions and on the number of literals needed to specify each precondition sub-case. Therefore, when the number of reactions N grows, the length of the PAs grows linearly.

There are two kinds of SSAs in the micro approach, one for the bond fluents and another one for auxiliary fluents. The SSAs for the bond fluents do not change when the number of reactions increases in KB. Therefore, the SSAs for the bond fluents stay the same regardless of N. However, assuming each reaction has M steps, we need on average $M/2$ auxiliary fluents for each reaction (since each split in a reaction requires introducing a new auxiliary fluent). Suppose that the context condition formulas in the SSAs for the auxiliary fluents need at most K literals, where K is a constant. Clearly, when N grows, the number of auxiliary fluents and the number of SSAs for auxiliary fluents grow linearly, but the lengths of the corresponding SSAs are independent of the number of reactions N.

3.2 Macro Approach

The other approach is to consider macro actions, meaning that each action in SC represents one full real chemical reaction. Our reactions are generic in the sense that they represent a class of individual reactions between similar molecules. This approach requires that all the atoms participating in a chemical reaction (all those atoms which change bonds) be listed as arguments in the action. In particular, this requirement accounts for stoichiometry of chemical reactions. For example, the ammonia NH_3 synthesis reaction $N_2 + 3\,H_2 \longrightarrow 2\,NH_3$ can be represented by the action term $hydrogenationOfNitrogen(n', n'', h_1, h_2, h_3, h_4, h_5, h_6)$, where the arguments n', n'' account for 2 nitrogen atoms, and $h_1, h_2, h_3, h_4, h_5, h_6$ account for 6 hydrogen atoms. As an example of actions in the macro approach, consider the ester water reaction (e_w_r action), which we write using long names of variables to suggest participating molecules:

$$e\_w\_r(carbOfEster_1, oxOfEster_2, hydOfWater_3,\\ oxOfWater_4, hydOfAcid_5, acidAnion_6)$$

This action represents the chemical reaction between an ester and a water molecules when an acid catalyst is present. We intentionally consider same reaction that was discussed before to illustrate the differences between the approaches.

It is worth noting that since splitting and forming of the bonds happen in a "package" in the macro approach, that is the chemical reaction is considered as a whole, there is no need to introduce auxiliary fluents to keep track of key atoms in intermediate chemical compounds. Each action will split and form the relevant bonds in one step. In other words, since there will be no intermediate state where a chemical reaction is in-progress, there is no need to keep track of intermediate key atoms. Consequently, this approach is convenient when order of bond formation and cleavage steps is incompletely known or when the mechanism of a chemical reaction should be treated as "black-box".

As usual in SC, we need to write a PA for each (re-)action. An example of a PA in the macro approach is the following:

$$\begin{array}{l} Poss(\ e\_w\_r(carbOfEster, oxOfEster, hydOfWater,\\ \quad oxOfWater, hydOfAcid, acidAnion), s\ ) \ \leftrightarrow\\ Carbon(carbOfEster) \wedge Oxygen(oxOfEster) \wedge\\ \quad Ester(carbOfEster, s) \wedge\\ \qquad Bond(carbOfEster, oxOfEster, s) \wedge\\ \quad Hydrogen(hydOfWater) \wedge Oxygen(oxOfWater) \wedge\\ \qquad\quad Water(oxOfWater, s) \wedge\\ \qquad\qquad Bond(oxOfWater, hydOfWater, s) \wedge\\ Hydrogen(hydOfAcid) \wedge Strong\_acid(hydOfAcid, s) \wedge\\ \qquad Bond(hydOfAcid, acidAnion, s), \end{array}$$

where $Water$ and $Ester$ are abbreviations. This example of a PA in the macro approach is simply describing when it is possible to perform the reaction between an ester and water: namely, we should have an ester molecule, a molecule of water and a molecule of strong acid. It also gives significance to the arguments of the action and defines each argument's role in the reaction. For example, the second argument gets significance because it is defined to be the oxygen that has a single bond with the key carbon atom of the ester molecule.

In the macro approach, there is one action per reaction. Therefore, the SSA for the $Bond$ fluent can be long, since we need to take into account the effects of all the (re-)actions on this fluent. Moreover, the SSA has to represent changes in bonds between all atoms participating in a reaction. For simplicity, assume we had only one action e_w_r. As a result of the e_w_r reaction, a few bonds will break and a few new bonds will be formed. The reaction is represented in Figure 3.

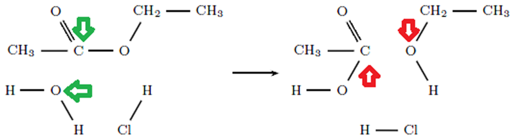


Figure 3: The ester and water reaction e_w_r produces an alcohol $CH_3{-}CH_2{-}OH$ and a carboxylic acid $CH_3{-}COOH$.

Let us explain the logic of SSAs in the macro approach using an example. Consider the partial SSA for the $Bond$ fluent:

$$\begin{aligned}&Bond(x, y, do(a, s)) \;\leftrightarrow\\ &\quad(\exists\, o, h, h', z)(\, a\!=\!e\_w\_r(x, o, h, y, h', z)) \vee (\ldots)\\ &\quad\vee Bond(x, y, s) \wedge\\ &\quad(\neg\exists\, h, o, h', z)(\, a\!=\!e\_w\_r(x, y, h, o, h', z)) \vee (\ldots)\,.\end{aligned}$$

The first line of the right hand side accounts for the bonds that will be formed as a result of the reaction. Recall that in the definition of e_w_r, the first and fourth arguments stand for the carbon atom of the ester molecule and the oxygen atom of the water molecule respectively, both marked with arrows on the left hand side of Figure 3. Thus, if the last action was e_w_r and x and y are the first and fourth arguments of the action, then there will be a bond between them in the situation that results from this reaction. Other combinations of arguments of an e_w_r action (or any other actions) that cause formation of bonds should be part of the disjunction on the first line to account for all bond formations.

The last line of the SSA accounts for the bonds that will split as a result of this reaction. Recall that the first and second arguments of an e_w_r action represent carbon of ester and oxygen of ester, respectively, both marked with the arrows on the right hand side of Figure 3. Thus, if the last action was e_w_r with x and y as first and second arguments, the $Bond$ fluent will become false for x and y, meaning that this bond splits in the next situation. Other combinations of arguments of an e_w_r action (or any other action) that cause a bond cleavage should be part of the disjunction on the last line of the SSA to account for all bond cleavages.

The middle line of the SSA accounts for the non-effect of the actions. If the last executed action occurs neither in the first part (bond formation), nor in the last part (bond cleavage), then there is a bond between x and y if only if they had already a bond in the previous situation, since the executed action has no affect on the bond between the atoms x and y.

As for scalability of axiomatization in the macro approach when the number of reactions in the KB grows, assume we have N reactions. Obviously, each reaction requires a PA, and formulation of each PA requires at most K (a constant number) literals, which can be fluents or abbreviations. Therefore, when N grows, the number of PAs grows linearly, but their length is independent of N. Additionally, each reaction affects at most M (a constant number) bonds. These effects have to be specified in the SSAs for the bond fluents. Therefore, the length of the SSAs for bond fluents has growth rate $O(N \cdot M)$, that is linear with respect to N.

4 Declarative Heuristics

In Section 6, we discuss what challenges can be addressed using our approaches, but in this section we concentrate on one of the discovery challenges - the CAOS problem. This problem can be cast in AI terms as the planning problem.

To solve the chemical organic synthesis problem, we need to find the right route from the set of initial states to one of the goal states. That is, we need to search for the correct route (sequence of actions) in the state space. In order to transform the initial state to the goal state we use a simple iterative deepening depth-first planning algorithm with declarative heuristics. It is important to avoid the useless actions that distract from progressing to the goal, and declarative heuristics are designed for this purpose. Declarative heuristics eliminate the useless parts of the search space and leave us with actions that are likely to reach our goal: a target molecule with desirable properties.

Two factors contribute to the termination time of a planning program. One factor influencing termination time is the time it takes to find in each situation what actions are possible, and another is the search for the right route to the target molecule from the starting materials. Two classes of declarative heuristics are introduced to help reduce the time needed to find the right route. Declarative heuristics are written using an abbreviation called $Useless(a, s)$, which holds if the action a is useless in the situation s. All that remains to be done is checking to see if the action a is useless or not, when considering the action as a possible continuation in s.

(1) Avoiding duplicate reactions. Imagine a situation where we have more than one instance of some molecule that can undergo a chemical reaction. For example, we know that esters and water react, and in our system we may have two identical ester molecules, and two molecules of water. In our set of the planning problems that we tried to solve, it was unnecessary to repeat the exact same chemical reaction (e.g., the ester-water reaction) if we had already taken that action once before.

In order to write declarative heuristics, we need to introduce new “marker” fluents that state whether an action has happened previously or not. In both micro and macro approaches, we need to write SSAs for fluents of this kind as well. Consider an example of this class of heuristic in the macro approach. Let the fluent $ExecutedEWR(s)$ hold if the e_w_r action was previously executed in the situation s. The SSA for this fluent is obvious. Then, the axiom

$$\begin{aligned}&Useless(\, e\_w\_r(c, o', h', o'', h'', x), s\,) \leftrightarrow\\ &\qquad\qquad\qquad ExecutedEWR(s)\end{aligned}$$

specifies that it is useless to re-execute the e_w_r action if we had already done so. As another example for the micro approach, consider the fluent $HasSplitAlcohol(s)$, which holds if a specific O$-$H bond in an alcohol was split in the situation s. Then, the axiom

$$\begin{aligned}&Useless(splitBond(o, h, 1), s) \leftrightarrow Hydrogen(h) \wedge\\ &\quad Alcohol(o, s) \wedge Bond(o, h, s) \wedge HasSplitAlcohol(s)\end{aligned}$$

states that it is useless to split the bond inside an alcohol molecule, if we had previously done so. For simplicity of presentation, this heuristic is written here with the implicit assumption that there is only one reaction that splits an alcohol molecule and therefore it is redundant to repeat it; this heuristic can be generalized.

(2) Avoiding reactions irrelevant to the taken route. In our set of experiments, it is useless to perform a chemical reaction that is irrelevant to the actions that have been taken so far. Reaction “A” is deemed to be *irrelevant* to reaction “B” if “B” does not use (directly or indirectly) the products of the reaction “A” in any way. Note that “A” might be irrelevant to “B”, but “B” might be not irrelevant to “A”.

The irrelevant reactions can be easily identified if we draw a graph of possible reactions (see Figure 4). In this graph, rectangles represent chemical compounds and ellipses represent chemical reactions. Edges going from compounds to reactions mean that the chemical compounds participate in the corresponding reactions. The edges going from chemical reactions to compounds mean that those compounds are products of the corresponding chemical reactions. The outgoing dotted edges in the graph mean that, as a result of the specific chemical reaction, one of the two (or more) compounds indicated by the dotted edges will be produced, but not both. A reaction "A" is irrelevant to reaction "B", if there is no path from "A" to "B". For example, in the sample graph of reactions depicted in Figure 4, if the "Alcohol-and-StrongBase" reaction were to generate a hydrogen molecule and not water, the "CarboxylicAcid-and-Base" reaction and the "Ester-and-Water" reaction would be irrelevant to the "Alcohol-and-StrongBase" reaction.

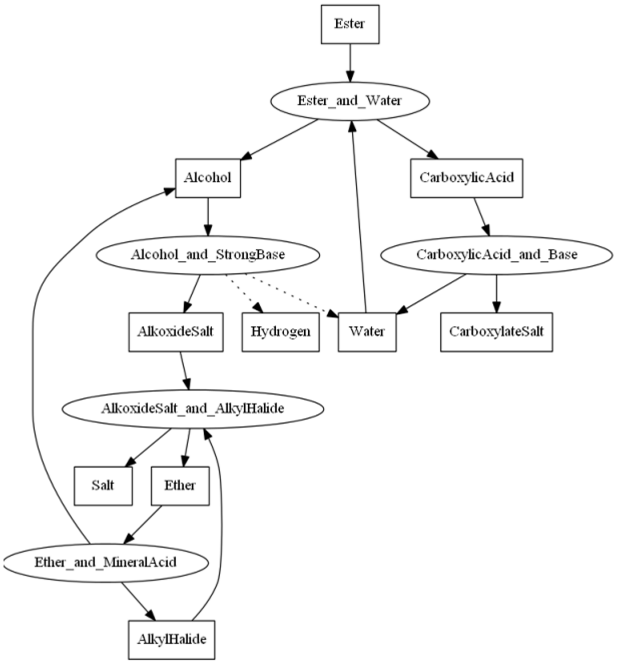


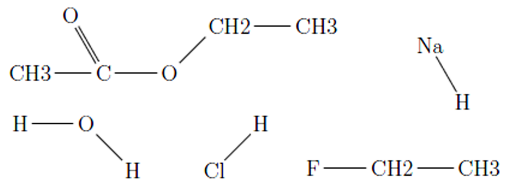
Figure 4: A graph of the reactions used in our program

Note that the lack of a path from the reaction "A" to reaction "B" means that "B" does not use any of the products of "A" in any way, neither directly, nor indirectly. This heuristic can be expanded so that a reaction "A" might be considered irrelevant to reaction "B", if the reaction "A" produced a product that was useful for reaction "B", but that product existed regardless of whether the reaction "A" occurred.

5 Implementation and Experiments

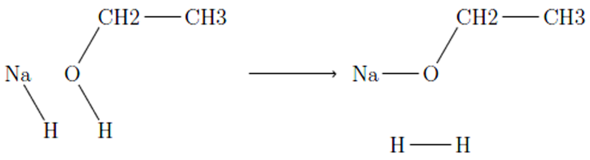
Both the micro and macro approaches have been implemented in ECLiPSe PROLOG. Subsequently, numerous planning problems corresponding to the graph in Figure 4 have been solved. The number of actions that are needed to complete a chain of chemical reactions depends on which approach is chosen (micro vs. macro). The micro approach requires noticeably more actions since one chemical reaction consists of several elementary actions of splitting/forming bonds between atoms. By contrast, in the macro approach, the number of actions constituting a desired chemical synthesis equals the number of chemical reactions in the chain.

In one of the planning problems that have been successfully solved, in the initial state, the existing molecules are an ester $R_1-COO-R_2$, a molecule of water H_2O, a molecule of hydrochloric acid HCl, ethyl fluoride CH_3CH_2F as the alkyl halide, and a molecule of sodium hydride NaH as the base. The chain of reactions consists of four chemical reactions. First (refer to Figure 3), the ester, water and hydrochloric acid react to produce an alcohol and a carboxylic acid. Second, the alcohol that was produced undergoes a chemical reaction with the sodium hydride and as a result, an alkoxide salt and a hydrogen molecule are produced. The third reaction is the reaction between the alkoxide salt and the ethyl fluoride which produces an ether and the salt sodium fluoride. The fourth reaction is between the ether that was previously produced and the hydrochloric acid, which is a mineral acid. This reaction produces an alkyl halide and an alcohol. Below, you can see the initial state given that the starting ester is ethyl acetate.

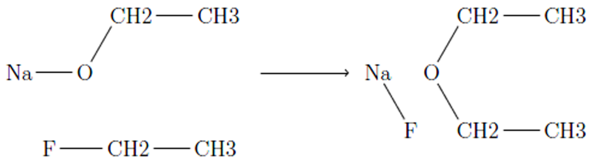


The first reaction was described in Figure 3. The following describes the rest of the reactions with the assumption of starting with ethyl acetate as the ester.

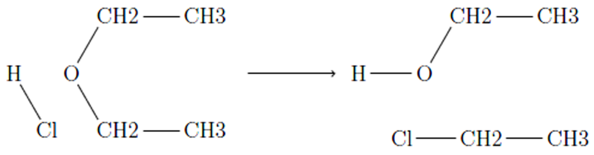
The 2nd reaction produces an alkoxide salt CH_3CH_2ONa:



The 3rd reaction produces an ether $CH_3CH_2–O–CH_2CH_3$:



The 4th reaction produces an alkyl halide $CH_3CH_2–Cl$:



The table below shows the time in seconds taken (on a desktop computer with 3.10GHz CPU and 4Gb memory) to discover a chain of 4 chemical reactions (which includes 18 micro actions) that creates an alkyl halide and an alcohol starting from an ester and a water molecule. The column on the left specifies which ester molecule was actually present initially. The ester molecules are sorted in the order of growing complexity.

	Micro (18 actions)	Macro (4 action)
Methyl acetate	1.30 sec	0.02 sec
Ethyl acetate	1.31 sec	0.03 sec
Methyl butyrate	1.28 sec	0.02 sec
Isopropyl acetate	1.56 sec	0.03 sec
Butyl butyrate	2.64 sec	0.09 sec

The results above were achieved without using heuristics in a simple environment, where there is only one instance of each molecule. In a more complex environment, where there are more than one instance of each molecule, it is prohibitive to use the program without heuristics. The table below shows how long it takes for the micro program without heuristics to solve a planning problem that requires the number of actions specified in the table, in an environment where there are two instances of each starting molecule.

Number of actions	Elapsed Time
1 action	0.02 sec
2 actions	1.37 sec
3 actions	606.47 sec

The same complex planning problem can be solved much faster when we take advantage of the heuristics explained above. The tables below confirm this fact in a similar environment with two instances of each of the starting molecules. For the micro approach, the dependency of time (in seconds) on the number of actions is the following:

#Acts	1	2	3	4	5	6	7	8	9	10	11	12
Time	0.03	1.73	3.24	5.12	7.36	10.56	14.57	18.74	23.48	28.84	48.13	41.96

For the Macro approach:

Number of actions	No Heuristics	With Heuristics
1 action	0.02 sec	0.06 sec
2 actions	0.02 sec	0.09 sec
3 actions	4.69 sec	5.09 sec
4 actions	152.47 sec	12.06 sec

These data show that, using a simple-minded planner with natural heuristics, our program can quickly discover nontrivial sequences of reactions by itself. Overall, this demonstrates the viability and potential of our SC-based logical representation of chemical reactions.

6 Discussion and Future Work

We demonstrated that SC can be used successfully to reason about reactions in organic chemistry. We illustrated our proposal with reactions involving single bonds between atoms, but reactions with double bonds (e.g., halogen addition reactions of alkenes) and with triple bonds (e.g., hydrogen addition reactions of alkynes) can be easily formulated as well.

So far in this paper, we focused on applying the situation calculus to solving the organic synthesis planning problem. It is clear that the automated organic synthesis is computationally intractable without heuristics when the number of reactions is large. However, our results indicate that with suitable domain dependent declarative heuristics, the program can quickly discover chains of reactions synthesizing the target compound. Also, the declarative heuristics can be supplemented with more traditional numerical heuristic functions. In (Vléduts and Geivandov 1974; Todd 2005), the CAOS problem is compared to playing chess: in both cases the computer programs have to search through a vast number of possible actions. Pursuing this analogy, one might expect that future organic synthesis programs enhanced with advanced heuristics will be able to solve practically relevant organic synthesis problems (Cook et al. 2012). Our logical framework represents an advance over the state of the art in CAOS for at least the following reasons. First, unlike previous CAOS systems that are procedural and lack formal logical semantics, our program benefits from an easy-to-understand declarative representation facilitating human comprehension and elaboration. Second, our logical representation is based on SC, which is well-explored in AI, and, as a consequence, it provides immediate connection with many related results in AI, so that CAOS can benefit from advancements in AI.

However, the proposed approaches are not limited to organic synthesis planning. They can be used to solve other computational problems as well (possibly, with some minor modifications), e.g., the *mechanism elucidation* problem. Mechanism elucidation roughly corresponds to determining the internal mechanism of a chemical reaction. It is different from the organic synthesis planning problem, because it cannot be understood as searching for a sequence of known chemical reactions. In contrast, mechanism elucidation requires finding a sequence of elementary bond splitting/forming operations that constitute a given reaction (informally, process structure of a single reaction has to be determined). It turns out that by utilizing a new set of PAs it is easy to adapt our micro approach to solve mechanism elucidation problems. Instead of formulating preconditions about intermediate molecules – knowledge about them is unavailable when mechanism is unknown – the PAs in the micro approach should allow execution of the elementary actions (bond formation or cleavage) under conditions that respect the rules of organic chemistry. For example, as long as constraints on valences of atoms (and other chemistry constraints based on the first principles) are not violated, the elementary actions should be possible. Thereby, the mechanism elucidation problem is reduced to the AI planning problem, where operators are splitting and forming bonds between atoms, the initial state is characterized by compounds before the reaction starts, and the goal state is a set of molecules produced by the reaction. Using this reduction, we implemented and solved the mechanism elucidation problem mentioned as an example in Section 3.5 of (Valdés-Pérez 1995). There has been a lot of previous research focused on scientific discovery through heuristic modeling of chemical reactions, e.g., the MECHEM program (Valdés-Pérez 1994; 1995). However, our proposal is different from MECHEM because we propose a more general framework based on SC for solving a variety of computational problems. The MECHEM program is focused only on mechanism elucidation problems, while the approaches in our paper have potential for solving other computational problems in addition to mechanism elucidation.

Another computational problem that can be solved is the *projection problem*. The projection problem consists in answering whether a given logical query formula holds after executing a sequence of ground actions. In chemistry, this means answering whether a formula characterizing a target molecule holds after a chain of chemical reactions. From computational perspective, solving the projection problem is much easier than solving a planning problem, since no search is required. The projection problem considered to be the basic reasoning problem in SC (Reiter 2001). It is worth noting that the projection problem is undecidable in general SC, if the initial theory D_{S_0} is an arbitrary first order logic theory. But recently, a fragment $\mathcal{P}$ of SC was introduced that guarantees decidability of the projection problem for the BATs conforming to $\mathcal{P}$ (Yehia and Soutchanski 2012). For the BATs that satisfy syntactic constraints in $\mathcal{P}$, the projection problem can be reduced to the satisfiability problem in a description logic $\mathcal{ALCO}(\mathcal{U})$, which is a fragment of a description logic underlying the Web Ontology Language (OWL2). Consequently, any off-the-shelf OWL2 reasoner can be adapted to answering the projection queries in $\mathcal{P}$. Fortunately, we recently found that our macro approach can be formulated in a lightweight extension of $\mathcal{P}$. Since description logics are not expressive enough to characterize cyclic molecules (e.g., molecules with rings), more expressive logic is required to represent them faithfully (Motik et al. 2009; Magka, Motik, and Horrocks 2012). Our future work includes exploring whether the projection problem can be solved for arbitrary complex molecules (e.g., including those with rings) by appropriately extending $\mathcal{P}$.

It did not escape our attention that the proposed approaches can be applicable in other areas beyond organic chemistry. In fact, our research was initially motivated by desire to develop a SC based representation for biological pathways that can be seamlessly integrated with existing semantic technologies. We conjecture that an extension of our logical language might be useful as a general powerful logical language for communicating biological pathways. We hope that continued successful research along the lines presented in our paper can have significant impact on improving scientific processes thanks to this new logical language for sharing biological pathways. We are going to explore how our logical framework can be integrated with existing languages for representation of biological pathways.

Future research plans include extending our logical theories to represent stereoisomerism and enantiomers. Also, we would like to further improve the performance of our programs by introducing new classes of heuristics and taking advantage of the recent KR research on modularity and decomposition.

We are looking for collaboration with scientists, in particular with experts in organic chemistry and in biological pathways. Interested researchers are welcome to contact us.

7 Acknowledgements

Thanks to the Natural Sciences and Engineering Research Council of Canada (NSERC) and to the Department of Computer Science of the Ryerson University for providing partial financial support. We are grateful to Dr. Shinsaku Fujita for sending us a copy of his book. Thanks to Dr. Gerry Zarnett for discussions of organic chemistry. The authors express gratitude to Dr. Eric Harley, Serhei Makarov, Alexander Orchard and anonymous reviewers for comments on the preliminary drafts of this paper.

References

Baader, F.; Calvanese, D.; McGuinness, D. L.; Nardi, D.; and Patel-Schneider, P. F., eds. 2003. *The Description Logic Handbook: Theory, Implementation, and Applications*. Cambridge University Press.

Chen, W. L. 2006. Chemoinformatics: Past, present, and future. *Journal of Chemical Information and Modeling* 46(6):2230–2255.

Cook, A.; Johnson, A. P.; Law, J.; Mirzazadeh, M.; Ravitz, O.; and Simon, A. 2012. Computer-aided synthesis design: 40 years on. *Wiley Interdisciplinary Reviews: Computational Molecular Science* 2(1):79–107.

Corey, E. J., and Cheng, X.-M. 1995. *The Logic of Chemical Synthesis*. Wiley.

Corey, E. J., and Wipke, W. T. 1969. Computer-assisted design of complex organic syntheses. *American Association for the Advancement of Science* 166(3902):178–192.

Fujita, S. 1986. Description of organic reactions based on imaginary transition structures. *Journal of Chemical Information and Computer Sciences* 26(4):205–242, and 27(3):99–120.

Fujita, S. 2001. *Computer-Oriented Representation of Organic Reactions*. Kyoto (Japan): Yoshioka Shoten.

Magka, D.; Motik, B.; and Horrocks, I. 2012. Modelling structured domains using description graphs and logic programming. In *Proceedings of the 25th International Workshop on Description Logics (DL-2012)*, volume 846.

Motik, B.; Grau, B. C.; Horrocks, I.; and Sattler, U. 2009. Representing ontologies using description logics, description graphs, and rules. *Artif. Intell.* 173(14):1275–1309.

Reiter, R. 2001. *Knowledge in Action. Logical Foundations for Specifying and Implementing Dynamical Systems*. MIT.

Todd, M. H. 2005. Computer-aided organic synthesis. *Chem. Soc. Rev.* 34:247–266.

Valdés-Pérez, R. E. 1994. Heuristics for systematic elucidation of reaction pathways. *Journal of Chemical Information and Computer Sciences* 34(4):976–983.

Valdés-Pérez, R. E. 1995. Machine Discovery in Chemistry: New Results. *Artif. Intell.* 74(1):191–201.

Vléduts, G. E., and Geivandov, E. A. 1974. *Automated Information Systems for Chemistry (in Russian)*. Nauka, Moscow.

Vléduts, G. E. 1963. Concerning one system of classification and codification of organic reactions. *Information Storage and Retrieval* 1(2):117–146.

Waldinger, R. 1975. Achieving several goals simultaneously. Technical Note 107, AI Center, SRI International. Reprinted in E. Elcock and D. Michie (Eds.), Machine Intelligence 8. Chichester: Ellis Horwood, 1977, pages 94-136.

Yehia, W., and Soutchanski, M. 2012. Towards an Expressive Practical Logical Action Theory. In *Proc. of the 25th Intern. Workshop on Description Logics (DL-2012)*, 389–399.