

# Design Patterns for the Generation and the Analysis of Chemical Reaction Networks

**Hugues Bersini**

IRIDIA – ULB – CP 194/6  
50, av. Franklin Roosevelt  
1050 Bruxelles - Belgium  
bersini@ulb.ac.be

## 1. Introduction

Chemical reactions among molecules rapidly create a complex web of interactions where it is nearly impossible by analytical means to predict which molecule will emerge in high concentration and which will not « survive » the network interactions. Among others, a lot of factors make this prediction quite delicate: The description of the molecules and the reaction mechanisms, the calculation of the molecular internal energy and accordingly the establishment of the reaction rate, the non-linearity of the reaction dynamics (made even more complicated by the order of the reaction). The reaction network evolves according to two levels of change called dynamics and metadynamics. The dynamics is the evolution in time of the concentration of the units currently present in the network. Their concentration changes as a function of their network interaction with the other units. The metadynamics amounts to the generation of new molecules by recombining chemical materials constituting the molecules existing so far in the network. An analytical approach is made hard by the continuous appearance of new variables in the set of the kinetic differential equations. Computer simulations could help a lot in facilitating the prediction of the structure of the emerging network as well as the nature of the “winning” and “loosing” molecules.

The model presented here is trying to capture in software, and in a very preliminary attempt, some scholar chemistry like the molecular composition and combinatorial

structure, basic reaction mechanisms as chemical crossover, bonds opening/closing, ionic reactions, and simple kinetics aspects as first or second order reactions. A lot of finer chemical aspects are still however left aside (which reaction takes place, which active group and which bonds are really involved, what is the value of the reaction rate,...) but in such a way that it should be easy for a more informed chemist to parameterize and precise the simulation in order to account for the influence of these aspects.

To make this parameterization and the interface with the chemist possible, Object Oriented (OO) type of computation appears to be the natural way to follow, at least today. The choice of object-oriented programming results from the attempt to reduce the gap between the computer modeling and its "real-world" counterpart. It is intrinsic to OO programming that by thinking about the problem in real-world terms, you naturally discover the computational objects and the way they interact. The Unified Modeling Language (UML) is a major opportunity for our complex system community to improve the deployment, the better diffusion and better understanding of the computer models, and especially when proposed to researchers not feeling comfortable reading programs.

The next section will present, through the UML class diagram, the basic architecture of the chemical reactor software. The main classes are «Atom», «Molecule», «AtomInMolecule», «Link», «Reaction» and all the sub-classes of possible reactions. Classical problems like the «canonization» of molecules will be just sketched in the following section. In the fourth section, some simulated reaction mechanisms will be presented while the fifth will sketch the complete chemical simulator and some preliminary results obtained in mineral and organic chemistry, departing from very simple molecules. These results will testify for the hardness in predicting the emerging molecules of the reaction network, and the benefits gained in resorting to this type of computer simulations.

## **2. The Chemical OO Class Diagram**

UML proposes a set of well-defined diagrams (transcending any specific OO programming language) to naturally describe and resolve problems with the high level concepts inherent in the formulation of the problem. It is enough to discover the main actors of the problem and how they mutually relate and interact in time to build the algorithmic solution of this problem. A simple and introductory overview of the UML language can be found in (Eriksson and Penker, 1998). However, by deliberately restricting our use of UML to the only class diagram, reader familiar enough with OO programming should not have any understanding problem. The main class diagram of the chemical computer platform is shown in figure 1.

The main classes necessary to represent the molecular structure as a computational graph are: «Molecule», «Atom», «Group», and their three ionic counterparts, «MolecularIon», «AtomicIon», «IonicGroup». Additional classes are «Link» and



object of the class molecule is not a single chemical element but rather the set (whose cardinality is the concentration) of all identical chemical elements. A molecular object remains one and only one specific object, with its concentration evolving as a result of chemical reactions and their specific rate. Molecules can be compared, and a set of methods is defined to describe the molecule or to retrieve part of it, such as atomic sub-groups, weak or strong links. Molecules, groups and atoms have their ionic counterparts with, as only difference, the additional presence of a “charge” attribute.

Molecules are graphs that are computationally structured (in a canonical form to be discussed in the next section) with nodes of class AtomInMolecule pointing to other atomInMolecule nodes. Each molecule possesses one and only one AtomInMolecule attribute called the *headAtom* and which can be seen as its “front door” (it would be the “1” in the molecule  $1(4\ 4\ 4\ 4)$ ). As soon as an atom enters into a molecule, it is transformed into an AtomInMolecule object. AtomInMolecule relates to atom since the identity of such an object is the same as its associated atom. Using natural recursive mechanisms, AtomInMolecules can be compared and duplicated to be part of new molecules (for instance to compose the molecular products of some reactions).

Finally, an object Link connects two atomInMolecule. It has a *number of bonds* and a given *energyToBreak* and a given *energyToOpen*, so that the weakest links are the first to break or to open in the reaction mechanism. For instance, two atoms of valence 4 will connect (to form a diatomic molecule  $4(4)$ ) with a link containing 4 bonds, and one atom of valence 4 will connect with four atoms of valence 1 ( $1(4\ 4\ 4\ 4)$ ), each link now containing one bond. Link objects intervene in the unfolding and the coding of the reaction mechanisms. For instance, one major method associated with the class Link is “*exchangeLink*” involved in crossover molecular reactions. Additionally, links can be opened and increase or decrease their number of bonds. In the model here, the energy of any link will only depend on the identity of the two atomic poles and the number of bonds. These energies are given at the beginning of the simulation

### *The Reactions*

Reaction is an abstract class that can only pave the way to derived or sub classes representing different specific reaction mechanisms like *crossover*, *openBond* and *closeBond*, *ionization*, *ionMolecule*, that will be illustrated next. Actually in our simulation, 20 sub-classes of Reactions inherit from the super-class. Common to all reactions is that they associate chemical reactants (molecule, molecularIon, Atom, AtomicIon) with chemical products (the same). In the simulations to be presented, chemical reactants and chemical products are at most two. Also reactions involve links. For instance, two links are exchanged in the crossover reaction and one link is just open to free some bonds in the openBond reaction. Among the several methods to be “concretized” in the derived classes, the “*doTheReactionFirst*” realizes the reaction for the first time, namely generates the molecular products, test their existence (to verify is they are new or already existing), calculates the rate (in a way to be explained next) and executes the reaction just once. The “*doTheReactionAgain*”

method just executes the reaction one time and boils down to modify the concentration of the molecular reactants and products according to the reaction rate. For obvious reasons of computational economy, an object reaction is first created and calibrated by the doTheReactionFirst method, then it is repeatedly executed by the doTheReactionAgain method

### 1.3 The Molecular Unique Computational Structure

Whatever chemical notation you adopt, for instance the “line-bond” or Kékulé, one way of reproducing the connectivity pattern of a molecule is by means of a computational graph. Here a symbolic linear notation perfectly equivalent to the graph, is adopted to describe the molecular computational graph. One example will be enough to understand it. Take the following molecule:

$$1(1(444)2(1(333))2(2(3))2(4))$$

“1” is an atom with valence 4, “2” is an atom with valence 2, and “3” and “4” are atoms with valence 1. The graphic tree version is given in fig.2.

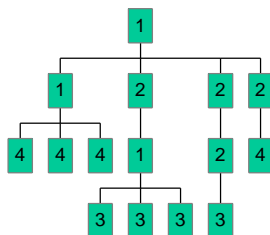


Figure 2: The computational tree structure for a molecule

In our linear notation, a “butane” molecule become:  $1(1(1(444)44)1(444)44)$   
 This string notation is automatically generated from the graph structure by a method being recursively executed through the graph. Cycles need to be detected by the method and are indicated by adding brackets at the two atomInMolecule which close the graph, like in the example below representing the benzene.

$$1[1](1(1(1(1[1](4(4(4(4(4))))))))))$$

In a first approximation, the connectivity shows symmetry both vertically and horizontally. The following rules need to be respected in order to shape the molecular graph in a unique canonical way:

**Vertically:** The highest node, i.e. the front door of the molecule (the initial “1” in our example of fig.2) must be the smallest of all the AtomInMolecule objects composing the tree.

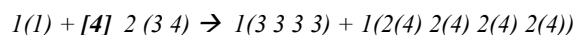
**Horizontally:** Below any node (i.e. any AtomInMolecule) the sub-nodes are arranged from left to right in an increasing order, the smallest to the left, the greatest to the right.

Clearly these two rules depend on the definition of “smaller” between two AtomInMolecule nodes. It is precisely defined in a way discussed in (Bersini 2000a and b). Organizing the graph in such a canonical way allows differentiating two structural isomers, i.e. molecules that contain the same number of the same atoms but in a different arrangement (like the chemical examples of the butane and the methylpropane molecules, both C<sub>4</sub>H<sub>10</sub> but with distinct connectivity patterns). However such a re-organisation can miss the differences still remaining between molecules showing a same connectivity pattern but with different spatial organisations i.e. the geometrical isomers.

#### 4 The different reaction mechanisms

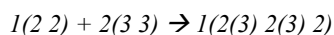
Several reaction mechanisms called “decomposition”, “combination”, “replacement” are repeatedly described in the chemical literature. Based on our syntactical definition of what is the molecular identity, a new nomenclature will be used here for the possible chemical reactions, with a simple illustration for a few of these reaction mechanisms. Every time a new molecular product is created as a result of the combination of two molecular reactants, this new molecule needs to be reshaped according to the canonical rules previously defined. Among the 20 reactions implemented in the code, there are:

*Single-link crossover:* the weakest links of each molecule are exchanged (suppose that “1” has valence 4, “2” has valence 2 and “3” and “4” have valence 1, and suppose the link “2-3” is weaker than the link “2-4”).

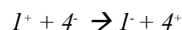


The bold values between brackets are *stoichiometric coefficients* needed in order to balance the chemical equations

*Open-bond reactions:* in the first molecule, a link containing i bonds opens itself to make j links of i/j bonds (here the first link “1-2” of the first molecule opens itself (i.e. frees one bond) and the first link of the second molecule breaks)



*Charge Transfer reaction:* it just amounts to the transfer of charges between two components



Each reaction is an object with (as attributes) at most two chemical reactants and two chemical products. Another important attribute to derive is the reaction rate. Suppose the following reaction:  $A+B \rightarrow C+D$ , the reaction rate  $K_{ABCD}$  is calculated as follows:  $K_{ABCD} = \alpha \exp(-E_{ABCD}/\beta T)$  where  $\alpha$  and  $\beta$  are two parametric attributes of any reaction. Their chemical meaning derive from the orientation and the shape of the molecules and influence the likelihood of the reaction.  $T$  is the temperature.  $E_{ABCD}$  is the activation energy of the reaction and, in a very naïve preliminary attempt, is calculated as follows. Every link of the molecule has a certain energy associated with it, which can be seen as the energy required to break the link:

$$E_{ABCD} = (\Sigma \text{ links energy broken in the reactant} - \Sigma \text{ links energy created in the product}) + \Delta \quad \text{if the first difference is positive}$$

$$E_{ABCD} = \Delta \quad \text{otherwise}$$

A reaction leading to a more stable molecule (i.e. when the difference is negative) needs to absorb much less energy to take place (just the energetic barrier  $\Delta$ ). This reaction is exothermic and much more likely and faster. Indeed the chemical kinetics (just first order reactions are considered) drives the concentration evolutions:

$$d[A]/dt = d[B]/dt = -K_{ABCD}[A][B]$$

$$d[C]/dt = d[D]/dt = K_{ABCD}[A][B]$$

Whenever the Reaction method “*doTheReactionAgain()*” is executed, a one time step integration of the four concentrations is achieved.

## 5 Results

In the final simulator, three vectors of objects are important: one containing all molecules, one containing all new molecules appearing at each time step of the simulation (and which will participate to create new reaction objects) and one containing all the reaction objects. In order to avoid the computational explosion that will rapidly occur as a consequence of the new molecules appearing in the simulation, molecules will enter a reaction only if their concentration exceeds a certain threshold. For similar reasons, only the weakest links of any molecules will be opened or broken for the reaction to occur.

Let’s now present two simulation results obtained, one for mineral and the other for organic chemistry. In the first one, the simulation is released with the following two molecules: 2(2) and 4(4), 2 is an atom with valence 3 (like N). After a great number of simulation steps the following molecules appear in the simulator: 2(444), 2(2(4)4), 2(2(44)2(44)2(44)), 2(2(2(4))44), 2(2(44)44), 2(2(44)2(44)4), 2(2(2(4))2(2(4))2(2(4))), 2(2(2(4))2(4)), 2(2(2(44)4)2(44)4), 2(2(2(2(4)))44) with their respective final concentration. The second simulation departs with one organic molecule: the benzene, and a second diatomic molecule 5(5) (it could be  $Cl_2$ ). An explosion of new organic molecules appear like: 1[1](1(1(1(1(1(1(1(1(1(1(1[1](4)4)4)4)4)4)4)4)4)4)4)4)4)4)4), 1(1(1(1(45)4)4)1(1(45)4)4),

