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

Hugues Bersini
IRIDIA – CP 194/6
Universite Libre de Bruxelles
50, av. Franklin Roosevelt
1050 Bruxelles – Belgium

Abstract

Chemical reactions among molecules rapidly create a complex and non-stationary 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. Computer simulations may help to realize these predictions. The work presented here aims at building software platforms that might play two roles. Following an adequate parameterization, they could be proposed to chemical or biological practitioners to model a particular system. They could also stand on their own and allow the discovery of generic laws characterizing the behavior of emergent complex systems. To make the parameterization and the interface easier, object oriented approach, by the use of UML diagrams, is the way to follow. In this paper the basic chemical classes of the model will be presented. Technical problems like the canonization of molecular graphs and the combinatorial explosion of the number of molecules will be faced. Some simulation results will be presented, departing from trivial diatomic molecules, but showing the difficulty to predict the emerging molecules and the benefits gained by resorting to computer simulations.

1 Introduction

Chemical reactions among molecules rapidly create a complex and non-stationary 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. To some extend a chemical reactor could be compared to an ecosystem. 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, and 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 [4] [7] [8] [10]. 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 work presented in this paper is a modest part of what the author understands to be the “Artificial Life” project. Alife aims at building computational platforms that might play two major roles. Following an adequate parameterization, they could be proposed to chemical or biological practitioners who could find in these platforms a helpful way to model a particular system. We then can speak of a set of computational design patterns, or software shell, which should easily be adapted to the simulation of a particular chemical or biological environment. These patterns must be sufficiently generic, understandable and easy to tune to represent the particular reality to target. On the other hand, these simulations can stand on their own and allow the discovery of generic laws, expressed in mathematics or linguistic terms, characterizing the behavior of emergent and complex systems. For instance, in Kauffman’s simulation of Boolean nets [13], some laws establish the stability or instability of generic networks as a function of the range of their connectivity. Others set the number of attractors as a function of the number of units in the network. In our specific case, these laws could connect, for instance, the number and the maximal size of possible molecules with the number of basic elements or as a function of the temperature.

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 or bonds opening/closing, and simple kinetic aspects as first or second order reactions. A lot of finer chemical aspects are still however left aside (which reaction takes place, 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 (00) type of computation is definitely the way to follow, at least today. The choice of object-oriented programming naturally follows from the attempt to reduce the gap between the computer modeling and its natural 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. In contrast, procedural programming forces the conversion of the real-world problem into the computer basic language. It is not OO programming languages (java, c++, smalltalk,) whose use is advocated here (although java contributes a lot to the spreading of chemical applets through the Internet). Everything programmed in an OO language can be alternatively done in a procedural way. It is rather the exploitation for communication ends of what OO languages, by increasing their use, have made more and more necessary, useful and prevalent in the computer world. That is, the development of high-level graphical notations to describe what the program is doing and how the reality is actually simulated without the need to resort to the code. The Unified Modeling Language (UML) is a major opportunity for scientific community to improve the deployment, the better diffusion and better understanding of the computer models, and especially when addressed to researchers not feeling comfortable reading programs.

The next section will present, through the UML class diagrams, 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, the three simulated reaction mechanisms will be presented while the fifth will present the complete chemical simulator. Chemical reactions rapidly lead to a combinatorial explosion of possible molecules on account of the growing number of molecules entering the reactions and the number of links that can be exchanged during these reactions. The sixth section will tackle this critical problem and show how it is possible to limit this explosion by common sense simplifications of the algorithm. The last section will present some preliminary results obtained departing from very simple and totally artificial diatomic molecules. These results will testify for the hardness in predicting the emerging molecules of the reaction network, and the benefits gained by resorting to this type of computer simulations.

2 The Chemical 00 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. It is beyond the scope of this paper to present UML although some of its symbols will be used to describe our chemical software environment. A simple and introductory overview of the UML language can be found in [9]. However, by deliberately restricting our use of UML to the only class diagram, readers familiar enough with OO programming should not have any understanding problem. Main parts of the class diagram will be now described.

2.1 Description of the molecular structures :

The main four classes necessary to represent the molecular structure as a computational graph are: « Molecule », « Atom », « Link » and « AtomInMolecule ». The Atom is the first basic class of the whole system. A fundamental attribute is the *valence*, which indicates in which proportion this atom will connect with another one to form a molecule. For instance an atom with valence 4 will connect with four atoms of valence 1 to form the molecule: *1(4 4 4 4*). The second major attribute is the *index*, which, in our simulation, relates to the value of the valence. Atom with a high valence will be given a small index value. This index simply needs to be an ordered index ("1", "2") for the canonical rules shaping the molecular graph to be possible. Since this index takes a unique value for each atom object, the way it is defined depends on what we take to be unique to any atom. Actually, in chemistry this index is given by the atomic mass.

The second basic class is the Molecule. As shown in the class diagram, molecule objects are compounds of atom objects. An attribute called *numberOfInstances* is a vector of integers whose elements are the number of times one specific atom appears in the molecule (i.e. four "4" and one "1" in the molecule *1(4 4 4 4)*). Molecules can be represented in a string linear form (in a way very reminiscent of SMILES [14]). *Concentration* is a very essential property of any molecule because, for obvious memory size constraints, an 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 objet 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.

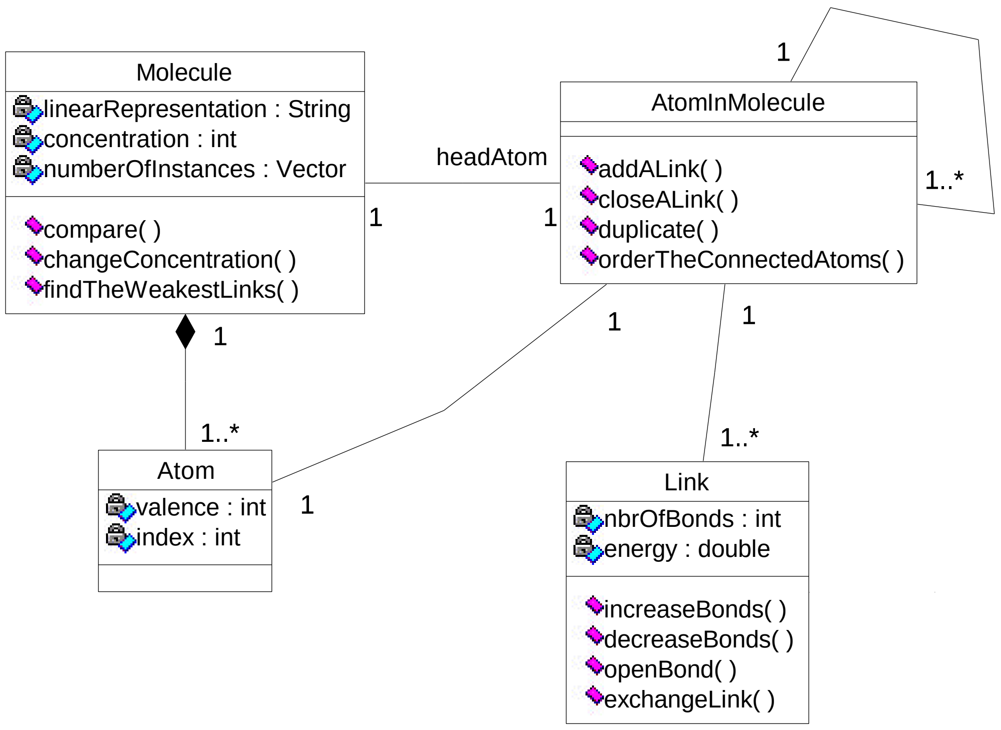

Fig. 1 The molecular part of the chemical UML class diagram

Molecules are graphs that are computationally structured (in a canonical form to be discussed in the next section) with pointers of class AtomInMolecule. Each molecule possesses one and only one AtomInMolecule pointer 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. It is a one-to-one relationship. AtomInMolecule are responsible for coding the graph structure of the molecule since they are associated with pointer attributes pointing in turn to a vector of AtomInMolecule objects. 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 given *energy* so that the weakest link is the first to break, and a *number of bonds.* 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 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 open and increase or decrease their number of bonds. In the model here, the energy of any link will only depend on the index of the two atomic poles and the number of bonds.

2.2 Description of the reactions

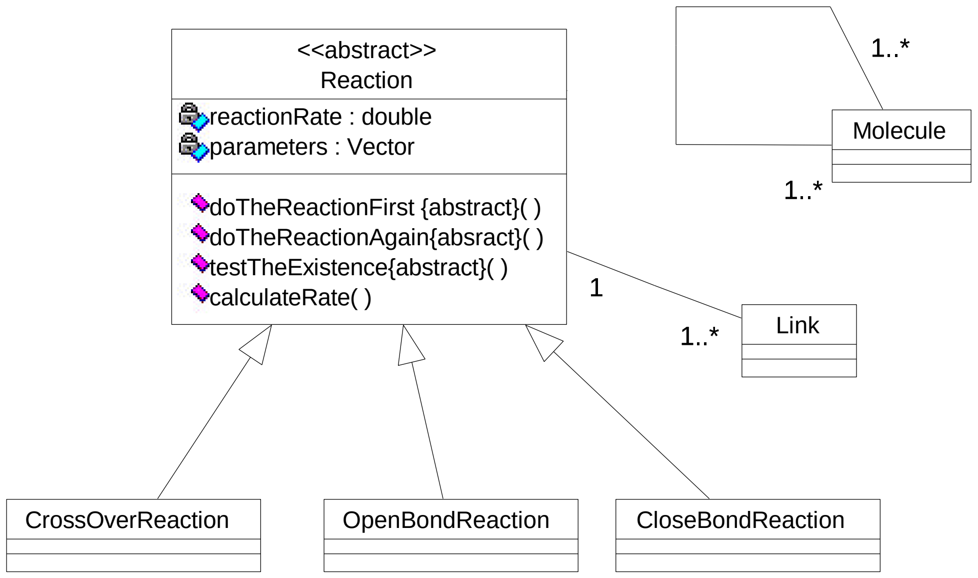

Figure 2: The Reaction part of the UML Class diagram

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, that will be illustrated next. Common to all reactions is that they associate molecular reactants with molecular products. In the simulations to be presented, molecular reactants and molecular 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 “concretize” in the derived classes, the *doTheReactionFirst()* realizes the reaction for the first time, namely generates the molecular products, tests 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 once. The *doTheReactionAgain()* just executes the reaction one time and boils down to modify the concentration of the molecular reactants and products according to the reaction rate. As a rule, a reaction is first created and calibrated by the doTheReactionFirst() method then it is repeatedly executed by the second method.

3 The Molecular Unique Canonical 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. For facility and space, the following linear notation will be adopted to describe a molecular computational graph. One example will be enough to understand it. Take the following molecule:

1 (1 (4 4 4) 2 (1 (3 3 3)) 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.3.

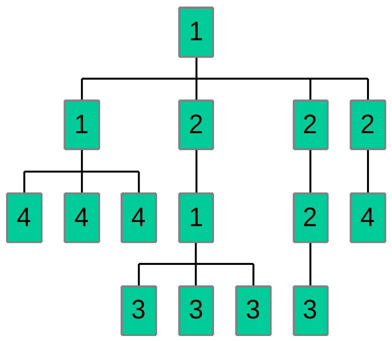

Figure 3: The computational graph structure for a molecule

In our linear notation, a "butane" molecule becomes:

C(C (C (H H H) H H) C (H H H) H H)

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.3) 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 defined in a way described in table 1 for two AtomInMolecule "n" and "m". You can see that the example given in fig.3 indeed complies with these rules: the highest "1" is connected first to a "1" then to a "2" whereas the other "1", although first connected back to a "1", is afterwards connected to atoms with higher index. The horizontal rule is equally verified for all connected atoms. Organizing the graph in such a 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 well-known chemical examples of the butane and the methylpropane molecules, both C_4H_{10} 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. These different spatial organisations can induce different optical properties, which is of no relevance for the remaining of the paper, but can also play a major role in the type of reactions these molecules are subject of.

Table 1: Which is the smaller between the
atomInMolecule n and m.

```
if (the index of n  < the index of m) {  the smaller is n  }
else
   if (the index of n = the index of m)
            { if ( n has no connected atom and m has no connected atom)   {the smaller is n}
            else
            if (the number of connected atoms of n > the number of connected atoms of m)
              {the smaller is n}
            else
            if ( the number of connected atoms of n = the number of connected atoms of m)
            {for all j connected atoms of n and m
                {if (the index of the jth connected atom of n < the index of the jth connected of m)
                {the  smaller is n ,
                break-the-loop}
                      else
                      { for all connected atoms of n and m
                                { redo recursively the same testing procedure}}}}
```

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 each reaction mechanism. 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 (transformed into its normal form (if borrowing Fontana’s words [12]).

- *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”).

  1(1) + ***[4]*** *2 (3 4)* → *1(3 3 3 3) + 1(2(4) 2(4) 2(4) 2(4))*

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

- *Multiple-link crossover*: several weak links are homogeneously exchanged between the two molecules.

  1(4 4 4 4) + ***[2]*** *2(2)* → *1(2 2) +* ***[2]*** *2(4 4)*

  The linear notation cannot account for the number of bonds characterizing the links. Here the 2-2 and the 1-2 links are double bonds.

- *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)

  1(2 2) + 2(3 3) → *1(2(3) 2(3) 2)*

- *Close-bond reactions*: it is the reciprocal of the previous reaction. It just needs one molecular product. In the first molecule, the two first links of one bond merge to form one link of two bonds:

  1(2(3) 2(3) 2) → 1(2 2) + 2(3 3)

- *Changing-valence reactions*: in the first molecule, the first atom of valence i increase or decrease its valence (here decrease)

  1(4 4 4 4) → 1(4 4) + 4(4)

- *Re-organizational reactions:* it is just a re-organization of the links in the molecule:

  1(1(2 3) 3 3 3) → 1(3 3 3 3) + 1(2)

This re-organisation is often accompanied by a change in the valence of one of the atom (here the "1"). In the simulations to be described in the following, only the single-link crossover, the openBond and the closeBond reactions will operate.

Each reaction is an object with (as attributes) at most two molecular reactants and two molecular products. Another important attribute to derive is the reaction rate. Suppose the following reaction: A+B->C+D, the reaction rate K_{ABCD} is calculated as follows:

$$K_{ABCD} = \alpha \exp(-E_{ABCD}/\beta T)$$

α and β are two parametric attributes of any reaction. Their chemical meaning derives from the orientation and the shape of the molecules and influences 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} = (Σ links energy broken in the reactant – Σ links energy created in the product) + Δ if the first difference is positive
$E_{ABCD} = \Delta$ 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 Δ). This reaction is exothermic and much more likely and faster. Indeed the chemical kinetics (just first order reactions are considered) drives the concentration evolutions:

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

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

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

In the simulation, the energy value associated with each link is close to real values (found in classical chemistry books for the O-O,H-H,C-C and N-N bonds [1] [2]): 1 == 1 (a link with four bonds), 1300, 1=-1 (a link with 3 bonds), 837, 1=1(a link with 2 bonds), 612, 1-1 (a link with one bond), 348, 2=2, 484, 2-2, 157, 3-3, 424, 4-4, 230, 3-4, 419, 1=2, 743, 1-4, 338, 1=2, 743, 1-3, 412, 2-3, 463, 2-4, 10. The calculation of the reaction rate presented here is nothing but a drastic simplification of the reaction mechanism. Nevertheless, this calculation, although in a much clever way, could still be derived from some properties of the molecules and the links constituting the reaction.

5 The Complete Chemical Simulation

The complete chemical simulation is sketched in table 2. Three vectors are important: one containing all molecules, one containing all new molecules appearing at each time step of the simulation, and one containing all the reaction objects.

Table 2: The Complete Chemical Simulation

```
At each time step:

For all existing reactions:
    Do the one time step reaction (method "doTheReactionAgain")
    Modify accordingly the concentration of the molecules involved
For all new molecules
    If the molecule concentration exceeds a given threshold
        Select the weakest links
         Create new close-bond reactions (method "doTheReactionFirst")
        Add them to the reaction vector
        For all new molecular products
            Transform them into the canonical form
            If the molecule already exists, just modify the concentration,
            Else:
            Add it to the new molecule vector

For all new molecules
  For all existing molecules
    If both molecule concentration exceed a given threshold
        Select the weakest links
        Create new open-bond and crossover reactions, add them to the reactions vector
        For all new molecular products
            Transform them into the canonical form
            If the molecule already exists, just modify the concentration,
            Else:
            Add it to the new molecule vector

Merge the new molecule vector with the molecule vector
```

In order to diminish the number of new molecules appearing in the system, molecules can enter a reaction only if their concentration exceeds a certain threshold. Also the selection of the weakest links for the reaction is done for similar reasons.

6 How to limit the combinatorial explosion

Since all molecules can enter in a reaction to generate new ones, together with the various reaction mechanisms and the number of links to potentially be involved in the reaction, it is easy to anticipate the computational explosion that will rapidly occur as a consequence of the number of new molecules to be continuously generated. This problem has been formally posed and analyzed in [11]. Here described in table 2, among others (discussed in [11]), two approximations have been proposed to limit the number of possible molecules appearing in the simulation. First only the molecules with sufficient concentration (i.e. exceeding a given threshold) will enter in a reaction to generate new molecular products. This limitation should not modify in a serious way the molecules that will finally dominate the reaction. In principle, this should stop a chaining of reactions whose last products slowly vanish. The no-effect of the elimination of poorly concentrated molecules is theoretically only meaningful in non-catalytic or autocatalytic systems. A small-concentrated molecule could still be important to catalyze other reactions.

The limitation to the weakest links to be involved in the reaction is strongly related with the way the reaction rate is calculated, as a function of the energy of the links of the molecules. A reaction breaking a strong link in a molecular reactant should occur much less likely than a reaction breaking a weak link. Thus the first reaction would lead to molecular products in a much weaker concentration than the second reaction. This simplification results to be an additional way to stop the chaining of reactions to produce marginal molecules.

This restriction to the weakest links, like the previous simplification, has the strong and undesired consequence of suppressing the reversibility of some chemical reactions. Suppose for instance a simple $A+B \rightarrow C+D$ crossover reaction. The reaction will be reversible only if two conditions are verified: 1) both C and D exceed the concentration threshold necessary to enter a reaction – 2) the weakest links to exchange in C and D are the ones that will give back A and B. In the following reaction:

3(4 5) + 3(6 7) → 3(3(7) 5) + 4(6)

The reaction will be reversible only if the link 3-3 is the weakest of the first molecule. Although suppressing the reversibility of some reactions is an undesired side effect of these simplifications, it is possible that the inverse reaction that would allow the reactants not to completely disappear could still be so unlikely that the reactant concentration at equilibrium remain negligible.

7 Preliminary Results

Let's take a first very simple example of a simulation departing from two elementary molecules 3(3) and 4(4) (remember that both 3 and 4 have valence 1) with the initial concentration set to 100. The main results of the simulations are the figures 4 and 5.

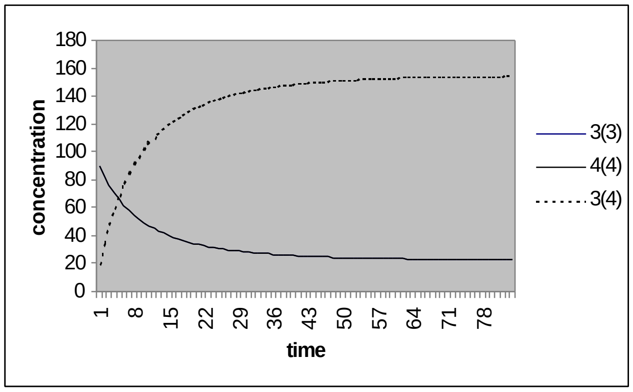

Figure 4: the concentrations in time of the reaction 3(3) + 4(4).

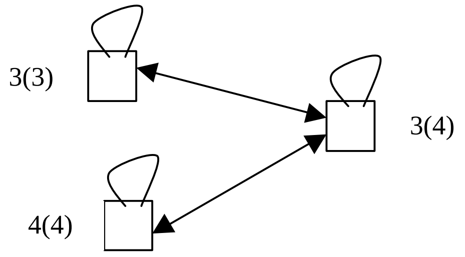

Figure 5: the three molecules and the reaction network connecting them.

In figure 4, you can see the concentration in time of the three molecules. The reaction is reversible, and since the difference in the energy of the links favors the production of the 3(4) (2*3-4 link is higher than the 4-4 + 3-3 links), the winner in concentration is not the most stable molecule, which is 3-3, but the most likely. In figure 5, you can see the reaction pathway network structure. Despite the triviality of the simulation, various reactions take place, indicated in the graph (whenever molecular reactants participate in a reaction producing a molecule: an arrow is drawn between the reactant and the product)

3(3) + 3(3) $\leftarrow$> [2]3(3)   this is shown on the reaction graph by the little loop closing on itself. The same for 4(4).

3(3) + 4(4) $\rightarrow$ 3(4) + 3(4)

3(4) + 3(4) $\rightarrow$ either 3(4) + 3(4)
or 3(3) + 4(4) allowing the reversibility

Now let's depart the simulation from again two molecules, but 2(2) and 3(3) instead. 2=2 is a two bonds link and 3-3 is a one bond link. Potentially the number of new molecules can explode in time due to the two bonds of the 2(2) molecule.

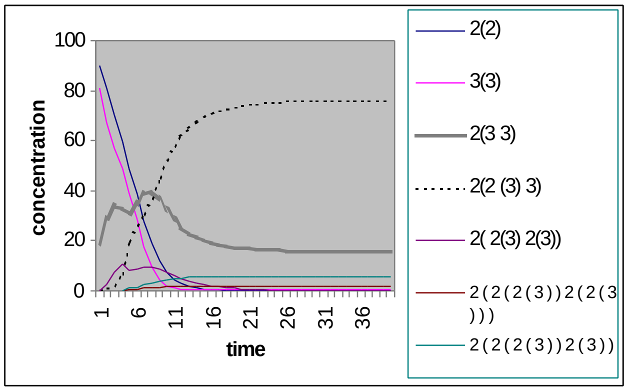

Figure 6: the concentrations in time of the seven molecules resulting from the reactions departing from 2(2) and 3(3).

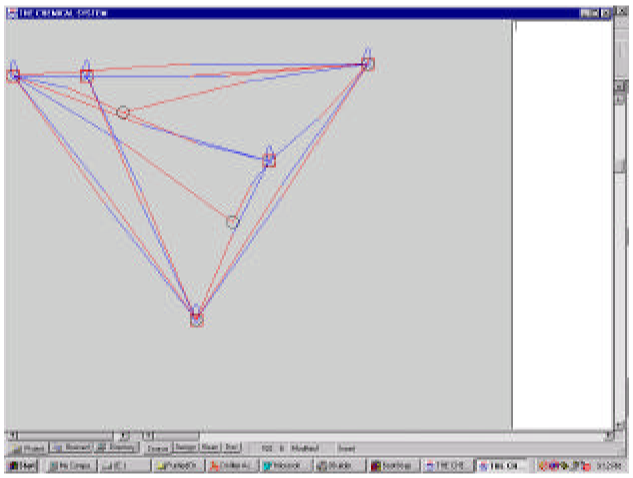

Figure 7: the seven molecules and the reaction pathway network connecting them.

To limit this number, the approximation strategies discussed above are applied. Only 7 molecules (the initial 2 + 5 new molecules) are created. The two graphic results are shown in figure 6 and 7.

The most concentrated molecules at the end of the run are 2(2(3) 3) and 2(3 3) which self-maintain by the reversible reaction:

$$2(2(3)\ 3) + 3(3) \leftrightarrow 2(3\ 3) + 2(3\ 3)$$

The first molecule is a rather surprising outcome since it is far to be the most stable. In terms of internal energy, 2(3 3) or 2(2) are more stable than 2(2 (3) 3) since the links 2-3 and 2=2 are much stronger than the link 2-2 (one of the weakest: 157). Moreover the molecule is asymmetric, another surprise, on account of the homogeneity of the reactions and the symmetry of the initial molecules. We are here in presence of a first example of a situation where the reaction pathway is counterbalancing the internal energy. Through the reactions allowed, 2(2) and 2(3 3) are much harder to produce than 2(2 (3) 3), which in contrast can be produced in several ways (like seen in the reaction pathway network of fig.5). Moreover, the asymmetry of the result is due to a form of "averaging" between favoring 2(3 3) for the internal energy and 2(2 (3) 2(3)) for the number of paths leading to this molecule. Finally, figure 6 shows how the molecule 2(3 3) takes the lead during the first part of the reaction to finally be transformed, through new reaction pathways, to the molecule 2(2(3) 3), for instance:

$$2(3\ 3) + 2(2\ (3)\ 2(3)) \rightarrow 2(2\ (3)\ 3) + 2(2\ (3)\ 3)$$

2(3 3), a more stable molecule than 2(2(3) 3), is first generated and rapidly grows in concentration until new molecules like 2(2 (3) 2(3)) enters the reaction and modifies the reaction pathway.

8 Conclusions

Without doubt, this paper has a richer computational than chemical content. All the chemistry addressed in the simulations is rather poor. The motivation is indeed to invite chemists to bring all the necessary chemical complements to improve the realism of the computational platforms. The design patterns presented in the UML form should be more attractive for them than pure code. These patterns, as chemically elementary as they are, could still participate in the skeleton of a much more realistic chemical software environment still to come. Seeing molecules as computational graphs is a rather classical attitude and more sophisticated canonization methods, accounting better for the chemical isomerism, could complete the active part of the definition of the Molecule and the AtomInMolecule classes. A lot of more realistic reaction sub-classes could be derived from the reaction superclass presented here. Still the molecular and link attributes will remain, but atomic sub-groups could be added, since a lot of reactions just involve specific active groups in the molecular reactants. Also the way the reaction rate is computed, an attribute derived from the properties of the molecule and the links associated with the reaction, could be a lot improved while remaining a derived attribute of the associated molecules and links. Finally the simplifications proposed to limit the combinatorial explosion could be much more chemically justified in favoring certain reaction pathways which are known to more easily take place.

Using UML to describe molecules and reactions is an effort to be compared with projects like CML [15] or SMILES [14], to express chemistry by understandable and semi-formal computer tools. It puts pressure on chemists who find some interest in these tools to clarify in computational terms some still fuzzy chemical notions, like the definition of the isomerism or the precise description of the reaction mechanisms. However the tools presented here must be seen as complementary since the emphasis is not on the coding of these computational objects but on the simulation of their behavior and interactions. Making prediction of chemical reactions, like for instance which molecules will be presented in high concentration once the reactions reach its equilibrium, is the challenge. Not only the software, once achieved a certain chemical realism, could help to predict the reaction outcomes, but like the "tracing" of an expert system reasoning, the reaction network graphs could help to explain why these molecules indeed emerge. Also this reaction network could be used to anticipate the effect of increasing the concentration of some of the molecules, and in general to anticipate the displacement of the equilibrium resulting from any perturbation.

REFERENCES

1. Atkins, P and L. Jones 1999: Chemical Principles – The Quest for Insight – Freeman.
2. Atkins, P. 1997: Physical Chemistry – Oxford University Press.
3. Bersini, H. (1999): Design Patterns for an Object-Oriented Computational Chemistry – In proceedings of the 5th European Conference on Artificial Life (ECAL'99) – Eds : Floreano, Nicoud, Mondada – Springer – Verlag - pp. 389-398
4. Bersini, H. (1999) The endogenous double plasticity of the Immune Network and the inspiration to be drawn for engineering artifacts – In "Artificial Immune System and Their Applications" – Springer Verlag – pp. 22-44.
5. Bersini, H. (2000): Chemical crossover – In Proceedings of the GECCO-2000 conference – pp. 825-832.
6. Bersini, H. (2000): Reaction mechanisms in the OO Chemistry – In Proceedings of the Artificial Life 7 Conference.
7. Calenburh, V., Varela, F. and H. Bersini. 1996. Immune idiotypic network – In International Journal of Bifurcation and Chaos – Vol. 6 No 9 – pp. 1691-1702.
8. Detours, V., Bersini, H., Stewart, J. and Varela, F. (1994): Development of an Idiotypic Network in Shape Space – Journal of Theor. Biol. (170), 401-404 (1994).
9. Eriksson, H-E, Penker, M. (1998): UML Toolkit – John Wiley and Sons
10. Farmer, D. 1991 – A Rosetta stone to Connectionism – In Emergent Computation, Forrest, S. (Ed.) MIT Press.
11. Faulon Jean-Loup 1999 - See http://www.cs.sandia.gov/jfaulon/MICS/kinetics/kinetics.html
12. Fontana, W. and L.W. Buss (1996). The barrier of objects: From dynamical systems to bounded organization in Casti, J. and Karlqvist, A., editors, Boundaries and Barriers, pages 56-116. Addison-Wesley.
13. Kauffman, S. (1993): The Origins of Order: Self-Organization and Selection in Evolution – Oxford University Press
14. SMILES – See http://www.daylight.com/smiles/ssmiles
15. CML – See http://www.xml-cml.org/