Large-scale simulations in Chemistry

Johan J. Lukkien

Computer simulations have become an important research instrument in chemistry and physics. Faster computers, larger memories and better algorithms allow more accurate simulations of increasingly larger systems. One particular area of interest is the simulation of catalyst behavior through discrete event simulation. At Eindhoven University we have developed software to perform this type of simulation. As these simulations require much processing time and memory, we have investigated the possibility to use a parallel machine.

Introduction

Over the past years the computer has become an indispensable research instrument in physics and chemistry, and, in fact, in the research of just any discipline. The role of computers can be summarized as in Figure 1: experiments (measurements), theory (analysis) and simulation are used to obtain a better understanding of a physical system.

Depending on the application domain, simulations can be based on first principles or on some abstraction (a model) of the physical world. In the case of first principles this model is the quantum mechanical description of the world. Ideally, we would like to simulate just any system from first principles but this clearly does not work. For example, simulating the flow through a pipe or the behavior of a car engine based on a quantum mechanical description is not possible. This implies that simulations of systems larger than several particles are based on some model, and that these models represent several levels of abstraction, each level abstracting from the details of the previous level. A contribution of simulation then is that it shows the overall behavior of the system while starting out from the model assumptions only. In this sense it gives insight in the relation between the microscopic specification (the model) and the macroscopic behavior (the observed overall behavior).

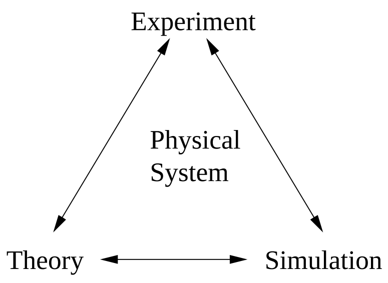
The size of the systems that can be simulated is determined by the speed and the memory limitations of the computing resources as well as the quality of the simulation algorithm. At any given time there is some maximum size on the system that can reasonably be simulated. Scaling up the simulations beyond that maximum requires either faster or more computing resources (i.e., a parallel computer). To what extent this scaling is possible depends on the complexity of the algorithm: if the algorithm has a bad time or memory complexity, the benefit of a faster or larger computer is not very significant. In addition, the used algorithm determines the effectiveness of parallelism.

In this article I will discuss as an example the work we did in the area of chemical simulations.

Modeling catalysts

In most chemical reactors used in industry, catalysts play a crucial role. A catalyst is a species that enables a certain reaction path, and, as a result increases the yield of a reactor. The catalyst itself is not modified in this process. A good example is the catalyst used in a car engine that is meant to support the oxidation of the poisonous exhaust gases CO and NO into CO2 and NO2. The catalyst in this case is a metal surface like platinum. It can

Figure 1: *Understanding physical systems is an interplay between experiments, theory, and simulations. Theory is developed based on experiments; computations and simulations are performed to verify theory; theory may be adjusted based on simulation results which may suggest new experiments as well, etc.*



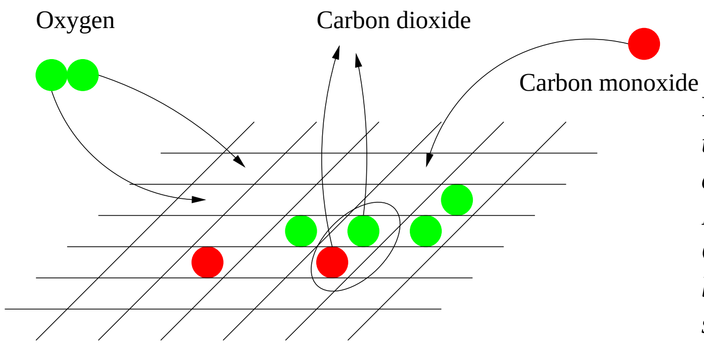


Figure 2: *A simple model for the CO oxidation on Platinum. It is assumed that both CO and O2 occur in the gas above the surface. Both CO and O2 may adsorb on the surface, O2 falling apart into two O atoms. A neighboring CO and O react to CO2 and leave the surface.*

be thought of as a regular lattice of sites that may contain particles. The effect of the lattice on these particles is that formation and breaking of chemical bonds becomes easier, and that particles 'meet' in a different way, viz., through being neighbors on the lattice. A simple model for the mentioned oxidation of CO on platinum was given by Ziff et.al., [1], see Figure 2. Although this model is a gross simplication of the physical system, it has some interesting properties and, because of its simplicity, it has been studied extensively. In a simulation of this system, the behavior over time is studied.

With this example in mind, it is straightforward to see what a general model for reactions taking place at a catalyst surface looks like. It comprises

- a set of particles,
- a discrete lattice of adsorbtion sites that each contain one particle,
- a collection of transitions, i.e., small, local modification of the lattice representing the reactions.

The reactions are also called *events*, an event being determined by its type (e.g., CO adsorption, O2 adsorption, etc.) and location. At any time there is a set of events that are possible, so-called *enabled events*. The evolution of the system boils done to the successive execution of enabled events. The order of the events is determined by a given probability distribution, which discribes for each event its time of occurrence. With this formulation we are in the area of *Discrete Event Simulation* (DES), a simulation technique that is known for quite some time and widely used ([3]). A simulation consists of the following steps

1. Store the enabled events together with a time of occurrence for each event.
2. Select the event with the minimal time and adjust the simulation time and the lattice.
3. Compute and store the new events that have become enabled and delete the events that have become disabled.
4. Continue from 2.

Although DES is quite well-established, its use is limited mostly to so-called *job-shop* and *work-flow* type of models. Our context is special in a number of respects.

1. For a simulation to be meaningful, the number of events should be at least several hundreds of thousands up to a billion or more.

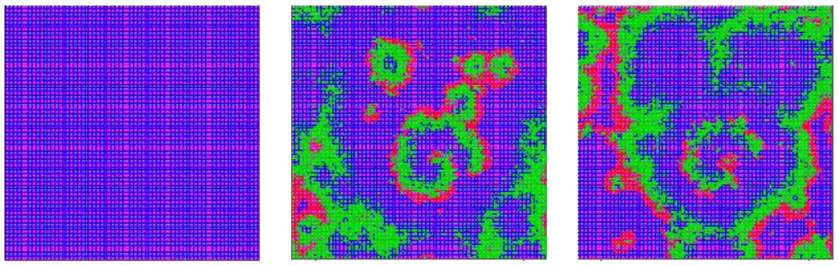


Figure 3: *Some snapshots during the simulation of a more advanced model for CO oxidation on platinum. The simulation shows the formation of patterns and it appeared that the system oscillates. The observed phenomena (e.g. pattern formation) are on a scale that requires lattices to be at least 1024x1024 sites. These particular snapshots were taken from a simulation that run for up to a month on a fast computer (an SGI Power Challenge) and used a little less than one gigabyte of memory.*

This number depends linearly on the size of the lattice.

2. Execution of an event disables (many) other events.

3. For larger systems, the list of enabled event may become extremely large. The size of this list is linear in the size of the lattice.

To understand the first point, the simulated time must be large enough to see the behavior of interest which usually implies the need to simulate many events. For the second point, the disabling can already be observed in Figure 2: for example, execution of O2 adsorption will disable any CO adsorption at the same position. Deleting these disabled reactions from the list is too costly. The result is that the list contains much garbage and uses much memory. Finally, going to larger and more complicated systems means extending the collection of reaction types and making the lattice larger. The latter is necessary because the size of the lattice must exceed the length scale of the observed phenomena, [4], see also Figure 3.

Approaches

Our focus has been primarily on improving the simulation algorithm in two respects: increasing the simulation speed and reducing the memory requirements. We indeed managed to find substantial improvements using characteristics of the model and, in particular, of the probability distributions ([5, 6]). On the one hand such an improvement is significant as it decreases simulation time and allows to indeed scale up the simulations. As a result, a simulation as in Figure 3 becomes feasible. On the other hand the improvement is only one or two orders of magnitude which, in due time, will also be reached by technology. Hence, the improvements are important for 'making tomorrows simulations possible today'. However, the most important contribution of this work as we see it is the development of a tool, CARLOS (see [2]), that takes a simulation model as input and simulates it with reasonable efficiency. The flexibility that such a tool offers makes it a lot easier to study different model assumptions and brings the analysis of models to a higher level.

Because of the interest in very large lattices we have also investigated the use of parallelism to speed up simulations. We have used the model of Figure 2 as a case study ([7]). Parallelism was introduced using a standard domain decomposition in which the lattice is divided across the available processors. Each processor has its own local simulator. When this similator discovers that a pattern falls across two (or more!) parts of the lattice it creates a new process on the respective processor to deal with this. This is illustrated in Figure 4. We used for the implementation ECL, ('Eindhoven Communication Library') which is a library developed in our group to support efficient communication between processors in a parallel machine. It is obvious that this efficiency will pertain particularly to process creation, since this occurs very frequently.

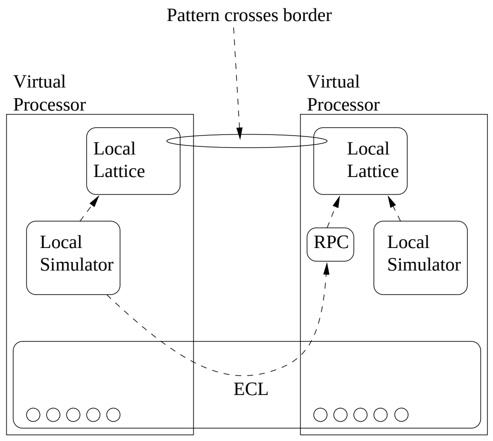


Figure 4: *Situation during simulation for two processors. The simulation is performed under control of a local simulator. When it needs to inspect or modify a pattern that crosses the border of the local lattice it creates a process ('Remote Process Call') on its neighbor to do this.*

The results from the case study were not very promising. First, if the strict sequential formulation of Discrete Event Simulation is to be maintained it is not really useful to have a parallel simulation as described above, because at any time only one event is executed. A part of the case study was to see if relaxing this requirement made a significant difference. This was done by synchronizing the processors at a larger time scale than at the scale of individual events. During these larger time steps, execution of events was not synchronized across processors. For this model, the error made in the simulation became visible only for small local lattices. The result was that indeed many processors can be used but that the local lattice of each processor should have a reasonable area to boundary ratio. The reason is that the area determines the amount of work while the boundary determines the amount of communication. The relative overhead of communication should be limited, hence the restriction. A second drawback was that the overhead through parallelism is significant and it required four processors before the parallel simulation had reached the same speed as the sequential one.

The conclusion is that in this case parallelism helps to go to larger systems; it does not help to improve the simulation of a relatively small system. Whether or not parallelism is generally applicable to this type of simulation is debatable, because of the error introduced by the course time steps in the domain decomposition. It certainly requires additional experimentation. For example, it is quite imaginable that the patterns in Figure 3 would look different if they would fall across different processors.

A more straightforward application of parallelism in this context is in collecting statistics. It is common practice to perform a simulation several times in order to get more reliable information out of it. These separate runs can be done independently and, therefore, in parallel. If pattern formation is not an issue then the size of the lattice in the simulation is determined mainly by the wish to reduce the noise on output parameters such as the evolution of concentrations and reaction speeds. This noise can also be reduced by performing several simulations with a smaller lattice and averaging the result. This cannot be done without limit of course, hence, the parallelism is limited here as well.

Conclusion

When looking at large-scale simulations, memory requirements and simulation speeds become an important issue and much effort goes into controlling these. For larger simulations the used algorithm is of crucial importance, in particular its complexity as this determines whether the simulator can take advantage of advances in technology. For the example of chemical reactions on a catalytic surface we have shown in a little more detail what this means. Parallelism may help to go 'one step further than todays technology'.

About the author

I received my masters degree in Mathematics in 1986 from Groningen University, the Netherlands. From that same university I received my PhD. in 1991. From 1989 to 1991 I worked at the California Institute of Technology. Since 1991 I have been with the Parallel Systems group at Eindhoven University, currently as an associate professor. My main interest has been the programming of parallel computers with an emphasis on two aspects: first, the actual realization and analysis of parallel programs for real-world applications and second, the methodological issues in this. The last few years I have been working on large-scale simulations in the area described in this article.

References

[1] Benjamin J. Brosilow, Erdogan Gulari, and Robert M. Ziff. Boundary effects in a surface reaction model for CO oxidation. *The Journal of Chemical Physics*, 98:674–677, January 1993.

[2] http://www.win.tue.nl/cs/pa/johanl/carlos

[3] I. Mitrani. *Simulation Techniques for Discrete Event Systems*. Cambridge University Press, 1982.

[4] R.J. Gelten. *Monte Carlo Simulations of Catalytic Surface Reactions*. PhD dissertation, Eindhoven University of Technology, 1999.

[5] J. J. Lukkien, J. P. L. Segers, P. A. J. Hilbers, R. J. Gelten, and A. P. J. Jansen. Efficient Monte Carlo methods for the simulation of catalytic surface reactions. *Physical Review E: statistical physics, plasmas, fluids, and related interdisciplinary topics*, 58:2598–2610, August 1998.

[6] J.P.L. Segers. *Algorithms for Simulating Surface Reactions*. PhD dissertation, Eindhoven University of Technology, 1999.

[7] John Segers, Johan Lukkien, and Peter Hilbers. Parallel monte carlo simulation of chemical reactions: A case study. In H. Liddell, A. Colbrook, B. Hertzberger, and P. Sloot, editors, *High-Performance Computing and Networking, Proceedings HPCN Europe 1996*, volume 1067 of *Lecture Notes in Computer Science*, pages 235–242. Springer, 1996.