Systems Self-Assembly

Natalio Krasnogor[1] and Marian Gheorghe[2]

[1] Automated Scheduling, Optimisation and Planning Research Group
School of Computer Science and Information Technology
University of Nottingham
Natalio.Krasnogor@Nottingham.ac.uk
[2] Department of Computer Science
University of Sheffield
M.Gheorghe@dcs.shef.ac.uk

Self-assembly is a process that creates complex hierarchical structures through the statistical exploration of alternative configurations. These processes occur without external intervention. The specific system that is self-assembled (from a given set of components) is determined by the way the statistical exploration of conformations is performed. In turn, the exploration mechanisms are constrained by the individual components that undergo self-assembly and the conditions imposed upon them by their local environment. Usually these constraints are related to the type of interactions in which the components engage. In general, components are autonomous, have no pre-programmed master assembly plan, and can only interact with their local environment and other components. Self-assembly is a powerful autopoietic mechanism whose power, as a reusable engineering concept, lays in the fact that it is distributed, non-necessarily synchronous, control mechanism for the bottom-up manufacture of complex systems. The control mechanism is distributed across a myriad of elemental components, none of which has either the storage or the computation capabilities to know and follow a master plan for the assembly of the intended system. Instead each component has a very limited behavioural repertoire which tells it what to do under a reduced set of well defined conditions.

Self-assembly processes are ubiquitous in nature. Understanding how nature produces self-assembly systems will represent an enormous leap forward in our technological capabilities. Self-assembly is an advantageous fabrication process because, with an appropriate set of components and associated interactions, these components will autonomously, robustly and efficiently assemble into a desired system. Robustness and versatility are some of the most important properties of self-assembling natural systems. The first of this two properties comes from the fact that usually these systems are composed of a large number of parts that can be interchanged and that can replace each other if one of them fails. On the other hand, versatility is given by the possibilities of re-configuring the way in which component parts relate to each other (i.e. there is a large degree of freedom in the way they interact). Additionally, the possibility of bulk manufacturing elemental components is attractive from a practical point of view as it cannot be expected that each component should be built independently. Bulk fabrication will ultimately make self-assembly an attractive concept for industry.

The purposes of nanofabrication, building nanostructures and nanoelectronic devices in chemical self-assembly has become an important avenue for employing and fabricating supramolecular nanostructures with, for example, useful electrical properties. Besides the modelling and the simulation of self-assembly in natural systems, self-assembly can be used in artificial systems as a powerful engineering principle to achieve a desired group effect or to form potentially autonomic structures exhibiting a hierarchy of emergent system properties. For example, a strategic research objective in robotics is to develop groups of robots (or micro-robots) which, having limited computation/communication capabilities, could self-assemble into a versatile and powerful robotic infrastructure. Another promising application is the development of autonomic, self-repairing, self-sustaining and self-healing software.

In [11] Reif says: ”We need improved software for designing novel DNA tiling assembles”.

Although major advances in the design of systems that exhibit self-assembly properties have been reported in the literature [10], much less has been said about the automated design of self-assembly. In [6] it is indeed tackled the problem of automated design of self-assembly for a very specific class of problems which are amenable to analytical solution. However it is unrealistic to expect that each and every system which self-assemblies through the bottom-up interaction of component parts will present properties which make them agreeable to a hand-made design. That is, we anticipate that in the near future, as the number of applications for self-assembly (and their complexity) will increase, a point will be reached where the humans cannot design the set of components and their interactions. A discipline of general systems self-assembly will thus require the analysis of computational and Kolmogorov complexities of the automated design of self-assembly and suitable algorithmic tools to deal with computationally hard cases. In [1] the complexity of self-assembled squares under generalised model of tile assembly was assessed. Several interesting results on the intractability of certain self-assembly processes were described. Although there are promises and limitations of specific self-assembly processes it is important to remark that NP-hardness results have not deterred the advance of other branches of science and engineering. On the contrary, NP-hardness results abound and are intrinsic to complex technology. A large variety of formulations that have been shown to be NP-hard, can be routinely solved by applying an arsenal of modern algorithmic techniques ranging from integer and linear programming, lagrangian relaxations to sophisticated metaheuristics like tabu search [4], simulated annealing [5], and memetic evolutionary algorithms [8]. Another avenue that needs to be developed is related to modelling aspects of autonomic self-assembly. Attempts have been made by using different formalisms like graph grammars [7], process algebra [3], but it is expected that new approaches (brane calculi [2], membrane computing [9]) will be also considered.

Bibliography

[1] L Adleman, A Goel, M Huang and P Moisset de Espanes. Running time and program size for self-assembled squares. In *Proceedings of the Annual ACM Symposium on Theory of Computing (STOC)*. ACM Press, 2001.

[2] L Cardelli. Brane Calculi - Interactions of biological membranes. *Proc. Computatinal Methods in System Biology.* Springer, 2004.

[3] V Danos and F Tarissan. Self assembling graphs. 2005.

[4] F Glover, E Taillard and D de Verra. A users guide to tabu search. *Annals of Operations Research*, 41:3-28, 1993.

[5] S Kirckpatrick, CD Gelatt and MP Vecchi. Optimization by simulated annealing. *Science.* 220(4598):671-680, 1983.

[6] E Klavins. Automatically synthesized controllers for distributed assembly: Partial correctness. In *Cooperative Control: Models, Applications and Algorithms*. Kluwer, 2002.

[7] E Klavins. Directed self-assembly using graph grammars. 2004.

[8] N Krasnogor, WE Hart and JE Smith. *Recent advances in Memetic algorithms.* Studies in Fuzziness and Soft Computing Series Springer 2004.

[9] G Păun. *Membrane Computing. An Introduction*. Springer-Verlag, Berlin / Heidelberg / New York, 2002.

[10] P Rothemund and E Winfree. The program-size complexity of self-assembled squares. In *Proceedings of STOC*, 2000.

[11] JH Reif. DNA lattices: A method for molecular-scale patterning and computation. *Computing in Science and Engineering Magazine, IEEE Computer Society,* 4(1):32-41, 2002.